NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

SCHOOL OF SCIENCES
DEPARTMENT OF INFORMATICS AND TELECOMMUNICATION

PROGRAM OF POSTGRADUATE STUDIES
" BIOINFORMATICS – BIOMEDICAL DATA SCIENCE "

MASTER THESIS

Using deep learning and natural language processing to predict protein-membrane interactions of peripheral membrane proteins

Dimitra E. Paranou

Supervisor: **Dr. Zoe Cournia,** Senior Researcher, Biomedical Research Foundation of the Academy of Athens (BRFAA)

ATHENS

MARCH 2023

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
" ΒΙΟΠΛΗΡΟΦΟΡΙΚΗ – ΕΠΙΣΤΗΜΗ ΒΙΟΙΑΤΡΙΚΩΝ ΔΕΔΟΜΕΝΩΝ "

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Χρήση βαθιάς μάθησης και επεξεργασίας φυσικής γλώσσας για την πρόβλεψη αλληλεπιδράσεων περιφερικών μεμβρανικών πρωτεϊνών με μεμβράνες

Δήμητρα Ε. Παρανού

Επιβλέπουσα: **Δρ. Ζωή Κούρνια,** Κύρια Ερευνήτρια, Ίδρυμα Ιατροβιολογικών Ερευνών Ακαδημίας Αθηνών (ΙΙΒΕΑΑ)

ΑΘΗΝΑ

ΜΑΡΤΙΟΣ 2023

MASTER THESIS

Using deep learning and natural language processing to predict protein-membrane interactions of peripheral membrane proteins

Dimitra E. Paranou

S.N.: 202200123

SUPERVISOR: **Dr. Zoe Cournia,** Senior Researcher, Biomedical Research Foundation of the Academy of Athens (BRFAA)

EXAMINATION COMMITTEE:

Dr. Zoe Cournia, Senior Researcher, Center for Translational Research, Biomedical Research Foundation of the Academy of Athens (BRFAA)

Dr. Theodore Dalamagas, Research Director, Information Management Systems Institute, ATHENA Research Center

Dr. Harris Papageorgiou, Research Director, Institute for Language and Speech Processing, ATHENA Research Center

March 2023

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Χρήση βαθιάς μάθησης και επεξεργασίας φυσικής γλώσσας για την πρόβλεψη αλληλεπιδράσεων περιφερικών μεμβρανικών πρωτεϊνών με μεμβράνες

Δήμητρα Ε. Παρανού

Α.Μ.: 202200123

ΕΠΙΒΛΕΠΟΥΣΑ: **Δρ. Ζωή Κούρνια,** Κύρια Ερευνήτρια, Ίδρυμα Ιατροβιολογικών Ερευνών Ακαδημίας Αθηνών (ΙΙΒΕΑΑ)

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ:

Δρ. Ζωή Κούρνια, Κύρια Ερευνήτρια, Κέντρο Μεταφραστικής Έρευνας, Ίδρυμα Ιατροβιολογικών Ερευνών Ακαδημίας Αθηνών (ΙΙΒΕΑΑ)

Δρ. Θεόδωρος Δαλαμάγκας, Διευθυντής Ερευνών, Ινστιτούτο Πληροφοριακών Συστημάτων, Ερευνητικό Κέντρο ΑΘΗΝΑ

Δρ. Χαρίλαος Παπαγεωργίου, Διευθυντής Ερευνών, Ινστιτούτο Επεξεργασίας του Λόγου, Ερευνητικό Κέντρο ΑΘΗΝΑ

Μάρτιος 2023

ABSTRACT

Peripheral membrane proteins play a crucial role in many biological activities: cell differentiation, proliferation, and communication with other cells, among other functions. Peripheral membrane proteins are regulated in various ways depending on their functions and cellular contexts, including binding to biological membranes. Characterizing interactions at the protein-membrane interface is crucial, as abnormal peripheral protein-membrane attachment is involved in the onset of many diseases. However, a limiting factor in studying and understanding protein-membrane interactions is that the membrane-binding domains of peripheral membrane proteins are typically unknown. Currently, the existing tools that predict the protein-membrane interface lack accuracy or/and are time-consuming. By applying Artificial Intelligence (AI) techniques in the context of Natural Language Processing (NLP), the accuracy and prediction time for protein-membrane interface analysis can be significantly improved. For instance, protein language models (pLMs) applied to protein structure prediction, offer a substantial advantage over other deep learning-based algorithms used for the same purpose, being 600 times faster with comparable accuracy. Hence, pLMs can be utilized to rapidly and accurately predict protein-membrane interfaces.

Herein, a machine learning methodology for predicting membrane-penetrating amino acids based on NLP and pLMs is described. First, we collect available data from two verified sources containing peripheral membrane proteins with experimentally known membrane-penetrating amino acids and generate features using pLMs to train machine learning and neural network classifiers. In the preliminary tests of this thesis, which employed default models without parameter optimization, the Multi-Layer Perceptron (MLP) demonstrated superior performance compared to other models, achieving the highest accuracy. Evaluation of the best MLP models, after fine-tuning the hyperparameters with Bayesian optimization, yields an F_1 score = 0.691 with Matthews correlation coefficient (MCC) = 0.652 and F_1 score = 0.622 with MCC = 0.577 for the two different pLM features respectively, for predicting correctly membrane-penetrating amino acids on unknown proteins of a test set. Close inspection of the results revealed that many of the false positive predictions are true positive since these amino acids are located adjacent to the protein-membrane interface. Thus, we defined a new cutoff radius around true positive membrane penetrating amino acids in order to include neighboring amino acids elevating the F1 score by an average of 0.11%.

To further improve the results, the attention heads of pLMs were extracted and used in two steps: 1. Investigate if the information about the membrane-penetrating amino acids exists in the hidden layers and the respective attention heads of these models, and 2. Use the attention maps to predict the protein-membrane interfaces. The first step verified that the information is indeed encoded inside the hidden layers of each pLMs. In the second step, logistic regression models were trained on the attention maps of each pLM for predicting the membrane-penetrating amino acids. Evaluation of the classifiers' accuracy produced an F_1 score = 0.366 with MCC = 0.344 and an F_1 score = 0.35 with MCC = 0.341 for the two different pLM attention maps, respectively, for the classification of amino acids that interact with the membrane.

Our MLP models trained in pLM features predict the protein-membrane penetrating amino acids in less than a minute for predicting the protein-membrane amino acids in all cases while other tools may require more than an hour. The generated MLP models provide highly promising results, yet with certain limitations that preclude generalization, namely the inability to make correct predictions for proteins outside the trained protein families. Conversely, the models trained using attention maps exhibited poor performance, which warrants further investigation. Overall, the results demonstrate the promising potential of using deep learning and pLMs to predict protein-membrane interactions faster and with similar accuracy compared to existing methods.

SUBJECT AREA: Computational Biology

KEYWORDS: machine learning, deep learning, neural networks, natural language processing, protein language models, embeddings, attention maps, peripheral membrane proteins, protein-membrane interactions

ΠΕΡΙΛΗΨΗ

Οι περιφερικές μεμβρανικές πρωτεΐνες παίζουν σημαντικό ρόλο σε πολλές βιολογικές δραστηριότητες: διαφοροποίηση κυττάρων, πολλαπλασιασμός και επικοινωνία με άλλα κύτταρα, μεταξύ άλλων λειτουργιών. Οι πρωτεΐνες αυτές ρυθμίζονται με διάφορους τρόπους ανάλογα με τις λειτουργίες και τα κυτταρικά τους πλαίσια, συμπεριλαμβανομένης της δέσμευσης σε βιολογικές μεμβράνες. Η κατανόηση των αλληλεπιδράσεων στη διεπιφάνεια πρωτεΐνης-μεμβράνης είναι ζωτικής σημασίας, καθώς η μη φυσιολογική σύνδεση πρωτεΐνης-μεμβράνης εμπλέκεται στην εμφάνιση πολλών ασθενειών. Ωστόσο, ένας περιοριστικός παράγοντας στη μελέτη και την κατανόηση των αλληλεπιδράσεων πρωτεΐνης-μεμβράνης είναι ότι οι περιοχές δέσμευσης μεμβράνης των περιφερικών μεμβρανικών πρωτεϊνών είναι τυπικά άγνωστες. Τα εργαλεία που υπάρχουν αυτή τη στιγμή για τη πρόβλεψη της διεπαφής πρωτεΐνης-μεμβράνης, έχουν έλλειψη ακρίβειας ή/και είναι χρονοβόρα. Με την εφαρμογή τεχνικών Τεχνητής Νοημοσύνης (AI), και πιο συγκεκριμένα της Επεξεργασίας Φυσικής Γλώσσας, ο χρόνος πρόβλεψης για την ανάλυση διεπαφής πρωτεΐνης-μεμβράνης μπορεί να βελτιωθεί σημαντικά. Για παράδειγμα, πρωτεϊνικά γλωσσικά μοντέλα που εφαρμόζονται στην πρόβλεψη της πρωτεϊνικής δομής, προσφέρουν ένα σημαντικό πλεονέκτημα έναντι άλλων αλγορίθμων βασισμένων σε βαθιά μάθηση που χρησιμοποιούνται για τον ίδιο σκοπό, καθώς είναι 600 φορές ταχύτερα με συγκρίσιμη ακρίβεια. Ως εκ τούτου, τα πρωτεϊνικά γλωσσικά μοντέλα μπορούν να χρησιμοποιηθούν για την ταχεία και ακριβή πρόβλεψη των διεπαφών πρωτεΐνης-μεμβράνης..

Στην παρούσα εργασία, περιγράφεται μια μεθοδολογία μηχανικής μάθησης για την πρόβλεψη αμινοξέων που διεισδύουν στη μεμβράνη με βάση την Επεξεργασίας Φυσικής Γλώσσας και τα πρωτεϊνικά γλωσσικά μοντέλα. Αρχικά, συλλέξαμε διαθέσιμα δεδομένα από δύο πηγές που περιέχουν περιφερικές μεμβρανικές πρωτεΐνες με πειραματικά γνωστά αμινοξέα που διεισδύουν στη μεμβράνη και δημιουργήσαμε τα χαρακτηριστικά χρησιμοποιώντας πρωτεϊνικά γλωσσικά μοντέλα για την εκπαίδευση ταξινομητών μηχανικής μάθησης και νευρωνικών δικτύων. Στις αρχικές δοκιμές, που χρησιμοποιήσαμε προεπιλεγμένα μοντέλα χωρίς βελτιστοποίηση παραμέτρων, το νευρωνικό Multi-Layer Perceptron (MLP) έδειξε ανώτερη απόδοση σε σύγκριση με τα άλλα μοντέλα, επιτυγχάνοντας την υψηλότερη ακρίβεια. Η αξιολόγηση των καλύτερων μοντέλων MLP, μετά από βελτιστοποίηση των υπερπαραμέτρων με χρήση της τεχνικής βελτιστοποίηση Bayes, αποδίδει βαθμολογία F_1 = 0.691 με Matthews correlation coefficient (MCC) = 0.652 και βαθμολογία F_1 = 0.622 με MCC = 0.577 για τα διαφορετικά των δυο διαφορετικών πρωτεϊνικών γλωσσικών μοντέλων ενός δοκιμαστικού συνόλου. Μια καλύτερη εξέταση των αποτελεσμάτων αποκάλυψε ότι πολλές από τις ψευδώς θετικές προβλέψεις είναι στην πραγματικότητα θετικές. Έτσι, μια νέα ακτίνα αποκοπής γύρω από τα πραγματικά θετικά αμινοξέα που διεισδύουν στη μεμβράνη για να περιλαμβάνει γειτονικά αμινοξέα αυξάνει τη βαθμολογία F_1 κατά μέσο όρο 11%.

Για περαιτέρω βελτίωση των αποτελεσμάτων, εξήχθησαν οι κεφαλές προσοχής των πρωτεϊνικών γλωσσικών μοντέλων και χρησιμοποιήθηκαν σε δύο βήματα: 1. Διερεύνηση εάν η πληροφορία σχετικά με τα αμινοξέα που διεισδύουν στη μεμβράνη υπάρχουν στα κρυφά επίπεδα και τις αντίστοιχες κεφαλές προσοχής αυτών των μοντέλων και 2. Χρήση

των χαρτών προσοχής για την πρόβλεψη των διεπαφών πρωτεΐνης-μεμβράνης. Το πρώτο βήμα επαλήθευσε ότι οι πληροφορίες είναι πράγματι κωδικοποιημένες μέσα στα κρυφά επίπεδα κάθε πρωτεϊνικού γλωσσικού μοντέλου. Στο δεύτερο βήμα, Logistic Regression μοντέλα εκπαιδεύτηκαν στους χάρτες προσοχής κάθε πρωτεϊνικού γλωσσικού μοντέλου για την πρόβλεψη των αμινοξέων που αλληλοεπιδρούν με τη μεμβράνη. Η αξιολόγηση της ακρίβειας των ταξινομητών παρήγαγε βαθμολογία F_1 = 0,366 με MCC = 0,344 και βαθμολογία F_1 = 0,35 με MCC = 0,341 για τους χάρτες προσοχής των δυο πρωτεϊνικών γλωσσικών μοντέλων αντίστοιχα, για την ταξινόμηση των αμινοξέων που διεισδύουν στη μεμβράνη.

Τα MLP μοντέλα μας που έχουν εκπαιδευτεί σε χαρακτηριστικά πρωτεϊνικών γλωσσικών μοντέλων προβλέπουν τα αμινοξέα που διεισδύουν στη μεμβράνη πρωτεΐνης σε λιγότερο από ένα λεπτό για την πρόβλεψη των αμινοξέων πρωτεΐνης-μεμβράνης σε όλες τις περιπτώσεις, ενώ άλλα εργαλεία μπορεί να απαιτούν περισσότερο από μία ώρα. Τα παραγόμενα μοντέλα MLP παρέχουν πολλά υποσχόμενα αποτελέσματα, αλλά με ορισμένους περιορισμούς που αποκλείουν τη γενίκευση, δηλαδή την αδυναμία να γίνουν σωστές προβλέψεις για πρωτεΐνες εκτός των εκπαιδευμένων πρωτεϊνικών οικογενειών. Αντίθετα, τα μοντέλα που εκπαιδεύτηκαν χρησιμοποιώντας τους χάρτες προσοχής εμφάνισαν χαμηλότερη απόδοση, γεγονός που δικαιολογεί περαιτέρω διερεύνηση. Συνολικά, τα αποτελέσματα αποδεικνύουν την πολλά υποσχόμενη δυνατότητα χρήσης βαθιάς μάθησης και πρωτεϊνικών γλωσσικών μοντέλων για την πρόβλεψη των αλληλεπιδράσεων πρωτεΐνης-μεμβράνης ταχύτερα και με παρόμοια ακρίβεια σε σύγκριση με τις υπάρχουσες μεθόδους.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Υπολογιστική Βιολογία

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: μηχανική μάθηση, βαθιά μάθηση, νευρωνικά δίκτυα, επεξεργασία φυσικής γλώσσας, πρωτεϊνικά γλωσσικά μοντέλα, embeddings, χάρτες προσοχής, περιφερικές μεμβρανικές πρωτεΐνες, αλληλεπιδράσεις πρωτεΐνης-μεμβράνης

To my family

ACKNOWLEDGMENTS

I would like to thank everyone who has contributed to the research described in this thesis. First, I would like to thank my supervisor, Dr. Zoe Cournia, for all the support, encouragement, and valuable advice. Her guidance was indispensable while I conducted research and wrote this thesis.

I would also like to thank the two members of my MSc diploma thesis examination committee, Dr. Theodore Dalamagas, Research Director at Information Management Systems Institute, ATHENA Research Center, and Dr. Harris Papageorgiou, Research Director at Institute for Language and Speech Processing, ATHENA Research Center for their advice and for their valuable time and feedback.

I would like to extend special thanks to Dr. Alexios Chatgizoulas for the guidance, the targeted suggestions, and the direction throughout the whole procedure of this thesis. In addition, I am grateful to all current and past members of Cournia lab for their help, suggestions in research, and fruitful discussions.

Last but not least, I would like to thank my family: my parents, my brother, and all my friends for supporting me spiritually throughout this thesis and in my life in general.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

PREFACE

The master thesis “Using deep learning and natural language processing to predict protein-membrane interactions of peripheral membrane proteins” has been conducted at the Biomedical Research Foundation Academy of Athens for the completion of the Postgraduate Program "Bioinformatics – Biomedical Data Science", Department of Informatics and Telecommunications, National, and Kapodistrian University of Athens, Greece.

The first chapter presents the motivation of the study, the importance of peripheral membrane proteins, and the current tools that exist in the literature for predicting protein-membrane interactions. Furthermore, the difficulties in studying the peripheral membrane proteins are listed and then the objectives of this thesis are defined.

In the second chapter, the theoretical foundations of the present work are described. First, machine learning theory is discussed, explaining the basic concepts of learning techniques, data processing and preparation, and the class imbalance issue that datasets have. Then, machine learning classifiers and their algorithms are described as well as optimization techniques are listed for improving the performance of the models. Moreover, the principal evaluation metrics for monitoring and evaluating the models’ outcomes are defined. Next, the concept of natural language processing is explained, and language models are described both within and beyond the realm of biology.

The results of the present thesis are presented in chapter three. First, the dataset collection is described in combination with the processing that followed concerning the preparation and annotation of data. In addition, the usage of language models for extracting the features is described and then, the outcomes of machine learning classifiers are listed. For the most promising classifiers, the optimization procedure accompanied by the predictions of the fine-tuned models is explained. Then, the usage of attention heads in the current task is determined, as well as the performance of models trained on these attention heads.

Finally, the epilogue of this thesis constitutes the conclusions along with the possible future perspectives of this study in Chapters four and five.

1. INTRODUCTION

Proteins are important blocks of life, playing essential roles in almost every biological process including structural support, enzymatic activity, transport, signaling, defense, and movement. They have a variety of characteristics, including a unique arrangement of amino acids, a three-dimensional form, and the capacity to communicate with other molecules. Proteins are involved in a variety of biological structures, such as the cytoskeleton, the extracellular matrix, the bones, and many others. Several essential functions for sustaining life, including, but not limited to, the transportation of molecules across cellular membranes, regulation of signal transduction pathways, and facilitation of metabolic reactions, are attributed to proteins [1]. It can be argued that proteins serve as a protective shield for genetic information and that the existence of life as we know it would be impossible without their crucial involvement.

1.1 Membrane Proteins

One of the most significant protein classes is membrane proteins. It is known that a third of the human proteome consists of membrane proteins and more than 60% of these are current drug targets [2]. Membrane proteins play a crucial biological role in the function of the cell, as they are responsible for a range of necessary processes for cell survival. These processes include the transport of ions and molecules, intra- and intercellular signal transmission, cell adhesion, and enzymatic activity [3]. Based on their structure, membrane proteins can be divided into transmembrane proteins, which are permanently anchored to or an integral part of the membrane, peripheral membrane proteins, which are temporarily and non-covalently attached to the membrane's surface or to other integral proteins, and lipid-anchored proteins, which bind to the membrane through a lipid molecule which is covalently linked to a specific amino acid residue in the protein (Figure 1) [4].

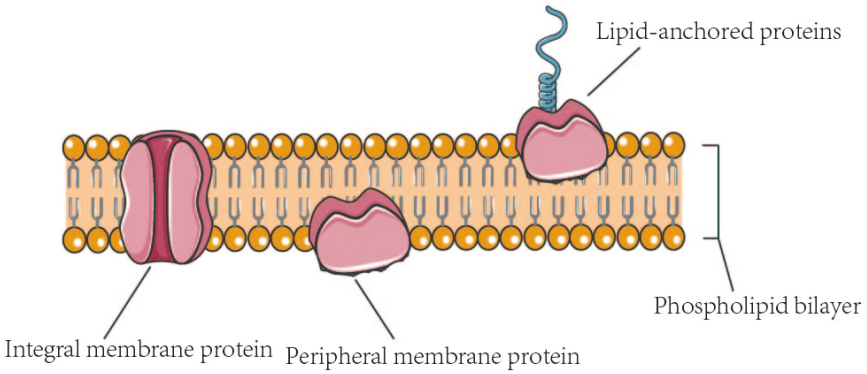


Figure 1: Classification of membrane proteins. Integral, peripheral, and lipid-anchored proteins are presented in a schematic way on a model cell membrane [5]**.**

1.1.1 Peripheral Membrane Proteins

Peripheral Membrane Proteins (PMPs) are responsible for a variety of biological functions, including signaling, recognition, membrane trafficking, cell division, and cell shape [6]. Some examples of PMPs roles are [4], [7]:

- *Enzymes*: Some PMPs act as enzymes, by catalyzing specific chemical reactions on the membrane surface.
- *Receptors*: Other PMPS are responsible for binding specific molecules and transmitting signals across the membrane.
- *Transporters*: Many PMPs act as transporters, helping to move molecules across the membrane.
- *Signal transduction*: Some PMPS are involved in signaling pathways that regulate cell growth, differentiation, and death.
- *Drug target*: A percentage of PMPs are potential targets for drug development, so studying their structure and function can have important implications for the treatment of diseases [8].

PMPs have the ability to alter membrane dynamics by binding to specific regions of the membrane and inducing local changes in its curvature, fluidity, or mechanical properties, and protein-protein interactions at the molecular level. These interactions are in reality very complicated, dominated by a variety of interactions, and have an interdependent impact on both the protein and membrane[9].

1.2 The protein-membrane interface

The abnormal attachment of proteins to the membrane is involved in overactivation or underactivation of peripheral membrane proteins and can result in the development of human disease (cancer, diabetes, etc.) [6]. Biological membranes are composed of various components that vary based on the type of cell or cellular compartment. The main components of biological membranes include proteins, lipids, and carbohydrates. These components are arranged in a double layer known as the phospholipid bilayer. The phospholipid bilayer is a typical structural component of cellular membranes, notwithstanding the variances. The hydrophobic (water-repelling) tails of the two layers of phospholipid molecules facing each other and the hydrophilic (water-attracting) heads facing outward make up the bilayer. This configuration aids in protecting the membrane's integrity and controlling how things enter and leave the cell.[10].

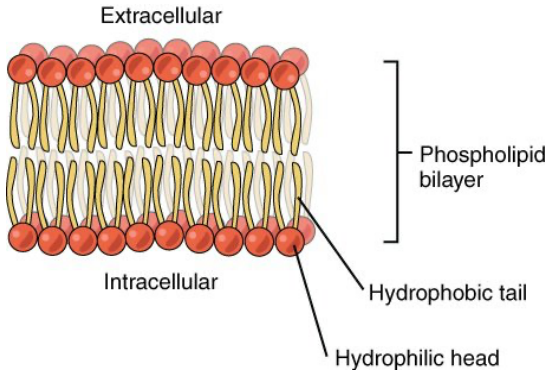


Figure 2: The phospholipid bilayer with its compounds

Through a series of distinct mechanisms, peripheral membrane proteins are drawn to and interact with cellular membranes. There are two primary categories of protein-membrane interactions, namely electrostatic and hydrophobic interactions. These interactions are governed by a complex energy landscape, which controls the specific and sometimes transitory interactions between a polypeptide and a bilayer. These principal interactions are described below:

- **Hydrophobic interactions**: The hydrophobic domains of peripheral membrane proteins can form hydrophobic interactions. A hydrophobic or amphipathic α-helix can be inserted into the membrane and form hydrophobic interactions with the phospholipid tails. In the same way, hydrophobic or amphipathic protein helices can interact with the lipid bilayer (Figure 3B & 3C) [11], [12].
- **Electrostatic interactions**: At the membrane surface, ion concentration gradients are caused by charged phospholipid head groups. Long-range electrostatic interactions between peripheral proteins and lipid headgroups result from charged membrane surfaces. Non-specific electrostatic interactions will cause even a partially positively charged protein to be drawn to a negatively charged membrane. The primary forces behind these interactions are the cationic amino acid residues of the protein (Figure 3A) [9].

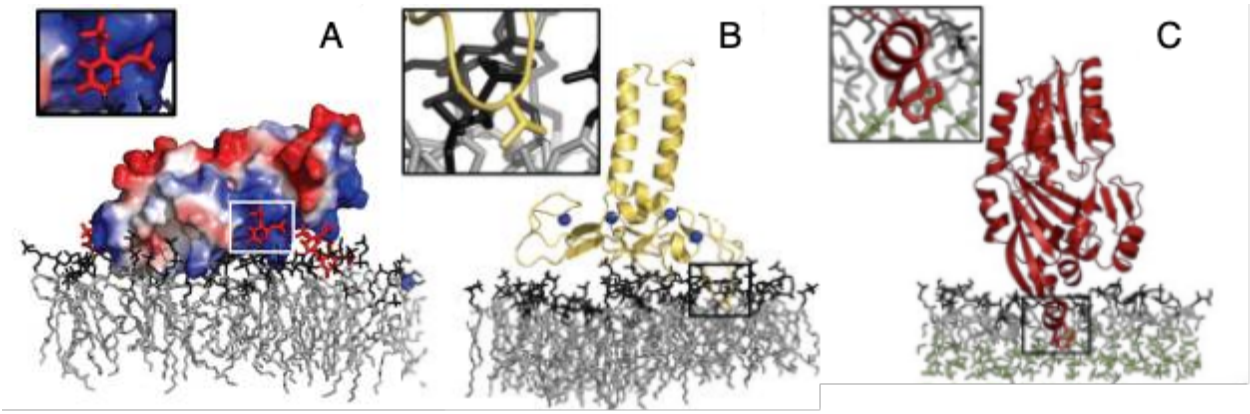


Figure 3: The two main types of protein-membrane interactions. Binding sites A) through direct electrostatic interactions, B) by inserting a hydrophobic loop in the hydrophobic core of the membrane, and C) by inserting a hydrophobic helix in the hydrophobic core of the membrane. Image adapted from [13].

Peripheral membrane proteins are also known to be multidomain proteins, with one or more domains to drive the protein-membrane binding [13]. As an example, Osh4 has been shown to have six membrane-binding domains (Figure 4), and it has been determined that protein binding happens because of random interactions with anionic lipids [14].

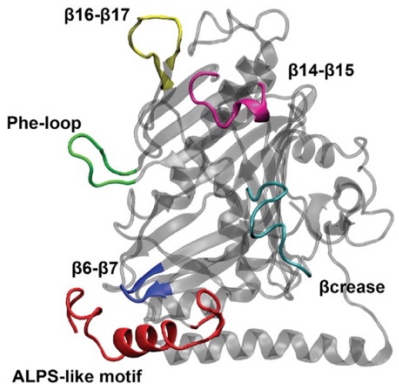


Figure 4: Binding sites of Osh4 [14]

1.3 Drugging the Protein-Membrane Interface

The transport of materials across the cell membrane, the activation of proteins and enzymes, and other cellular processes can all be influenced by peripheral membrane proteins. Disrupted cellular pathways and pathological conditions can arise from the overactivation or underactivation of peripheral membrane proteins, as well as abnormal binding of proteins to the membrane due to mutations in the membrane-binding domain. Consequently, modifying protein-membrane interactions presents a novel therapeutic strategy for several disease indications, particularly in the context of targeting membrane proteins that were previously considered undruggable [4]. Some of the most well-known pharmaceutical research targets are: the KRAS protein, which is said to be one of the most common oncogenic gene drivers in specific human cancers [15], [16], such as pancreatic cancer and for which a drug was recently developed [17], PI3Kα, which is one of the most frequently over-activated kinases in solid tumors [18], the CD73 enzyme that is implicated in Systemic lupus erythematosus [19] and tumors [20], and many more proteins [21], [22].

The presence of cavities within the membrane-binding domain of peripheral membrane proteins that can be targeted by drugs underscores the potential of targeting the protein-membrane interface [23]. Apomyoglobin was employed in the initial research investigating the mechanism by which a cytosolic protein interacts with membranes.[24]. These studies established that apomyoglobin interacts with membranes in a pH-dependent manner, with pH-dependent unfolding exposing areas of the protein that can bind to and interact with lipid membranes. In general, it is known that seven are the major classes of membrane binding domains, C1, C2, PH, FYVE, PX, ENTH, and BAR [25].

In addition, there are examples in the literature that report the ability to drug the protein-membrane interface, by verifying the binding of small molecules to membrane-binding domains and inhibition of protein-membrane interactions [26]. A notable inhibitor that was

designed for BRAG2 protein, managed to bind in the PH domain and inhibit allosterically the protein, without disrupting the protein-membrane interactions [27].

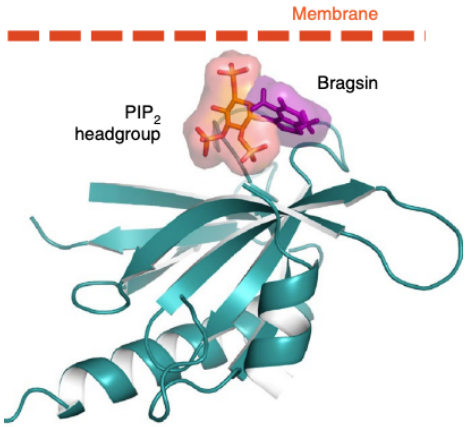


Figure 5: The BRAG2 protein (cyan) with the Bragsin inhibitor (purple) and the lipid PIP2 (orange). The inhibitor binds to the PH domain of BRAG2 and does not disrupt the protein-membrane interaction [27]**.**

Regarding the physicochemical characteristics of the interface between proteins and membranes, it is worth noting certain properties that stimulate the interaction of peripheral membrane proteins with the membrane. These properties are crucial for investigating these proteins as potential targets for drug development.

Protein-lipid interfaces exhibit specific chemical and topological features that are distinct in nature, such as amphipathic alpha-helices flanked by flexible hinge or loop sections, solvent exposure areas, or the presence of cationic patches surrounding aromatic and aliphatic areas that bind to the negatively charged bilayers [9], [28] that are frequently found in the inner leaflet of the plasma membrane [29]. Consequently, two factors that influence protein-membrane closeness and protein anchoring to the hydrophobic fatty acid tails of the lipid bilayer must be taken into account: a) long-range electrostatic interactions that promote protein-membrane proximity; and b) hydrophobic interactions (Figure 6) [29].

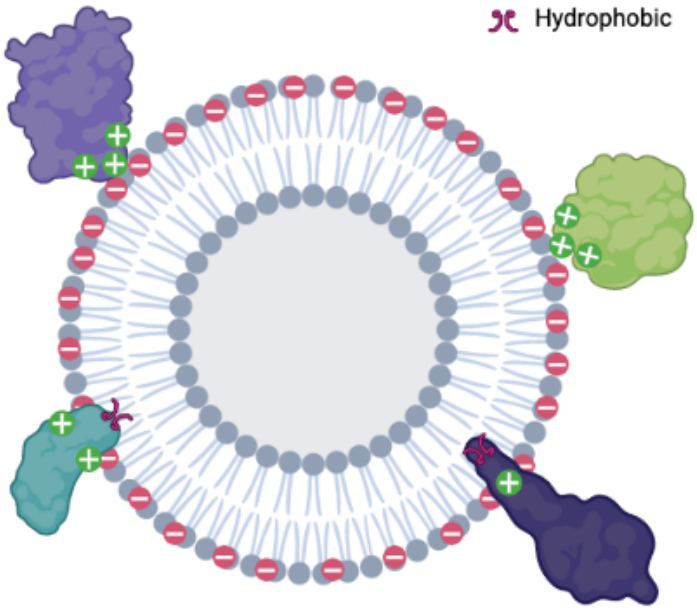


Figure 6: Representation of the protein-membrane anchoring. Hydrophobic interactions subsequently secure the protein to the hydrophobic fatty acid tails of the lipid bilayer after electrostatic interactions propel the protein into the membrane.

Proteins belonging to the peripheral membrane category, temporarily and partially bind to the membrane and as a result, only a tiny piece of the protein interface at the membrane-interface is fully inserted into the membrane. The fact that the protein interface that interacts with the membrane is solvent-exposed is thus an essential physicochemical property of the protein-membrane interface [30]. Additionally, the charge is a crucial physicochemical aspect of the protein-membrane interface since the two outer membrane layers are typically negatively charged. Lysine, arginine, and histidine amino acids are typically found in large groups on the protein surface of peripheral membrane proteins, where they interact with the membrane through long-range electrostatic interactions [9]. Protein-membrane sensors are driven by long-range electrostatic interactions, but these interactions are weak and insufficient to bind the protein to the membrane.

The biological membrane is amphiphilic, which means that its two leaflets that come into contact with water-based environments are hydrophilic, and its core is hydrophobic. Through hydrophobic interactions and owing to the amphiphilic character of the membrane, peripheral membrane proteins attach to the membrane, by introducing hydrophobic amino acids into the membrane's hydrophobic core. Hydrophobic amino acids typically end up buried deep inside the proteins due to the favorable nature of hydrophobic-hydrophobic interactions [31], [32]. The hydrophobic amino acids of peripheral membrane proteins that are exposed to the solvent can frequently interact with the hydrophobic core of the membrane, leading to the formation of protein-membrane associations. Alternatively, they can also interact with hydrophobic amino acids that are exposed to the solvent of other proteins, resulting in the formation of protein-protein interactions that minimize energy expenditure [33]. Some amino acids prefer to locate in the hydrophobic core of the membrane instead of the water-based environment. The amino acids that localize on the water-membrane interface are alanine, arginine, asparagine, aspartic acid, glutamic acid, glycine, charged histidine, lysine, and serine,

while those that prefer the hydrophobic core are isoleucine, leucine, methionine, phenylalanine, proline, and valine. The amino acids that prefer a border region between the water-membrane interface and the hydrophobic core are cysteine, glutamine, threonine, tryptophan, and tyrosine [34]. All the above were calculated by the free energies ΔG (kcal/mol) of transfer for each amino acid from water to phosphatidylcholine interface and n-octanol for each amino acid, resulting in the residue interface and octanol hydrophobicity scales (Figure 7).

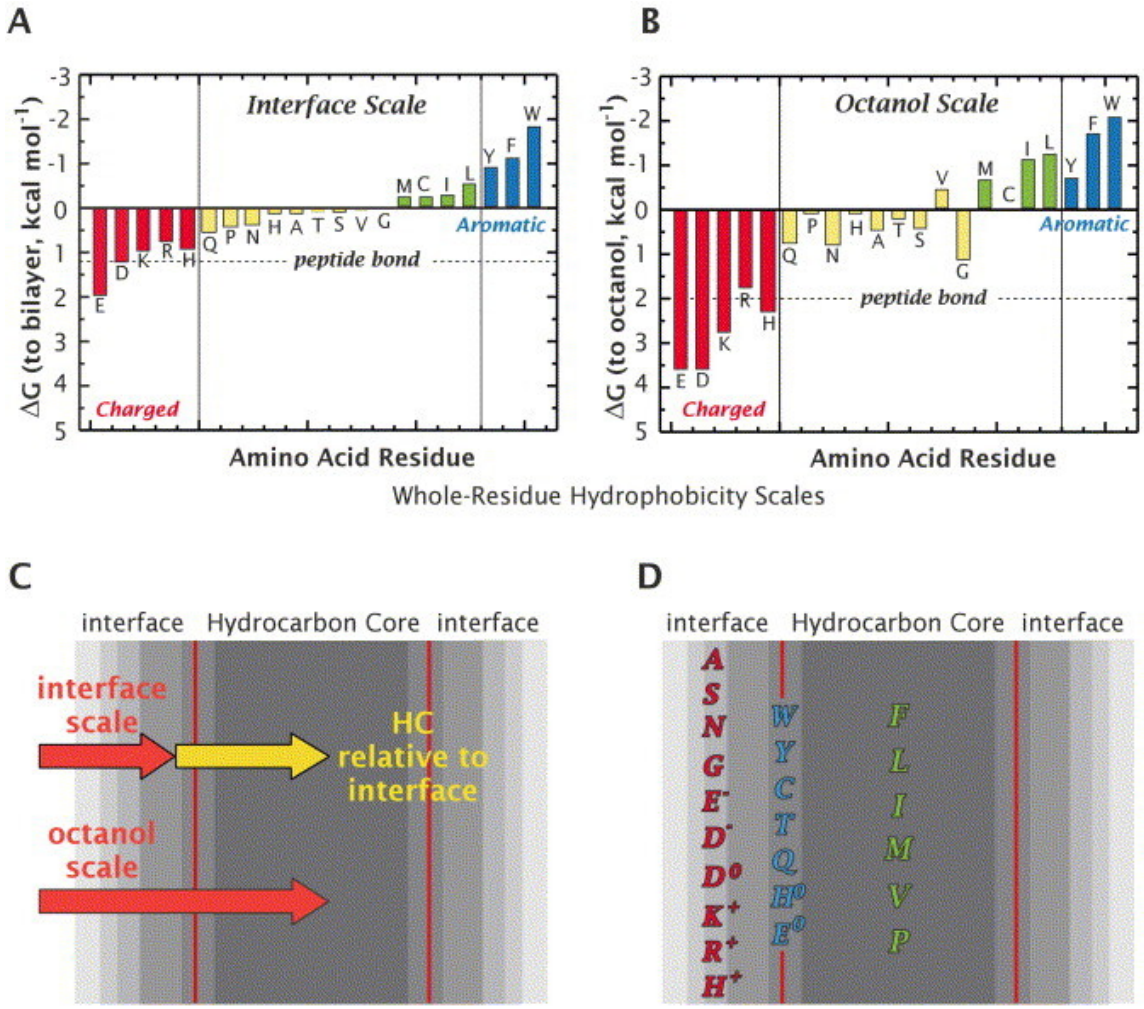


Figure 7: A) Wimley-White interfacial hydrophobicity scale. B) Wimley-White octanol hydrophobicity scale. C) The basis for deriving the octanol-interface scale. D) The relative amino acid positions in the membrane are based on the octanol-interface scale [34].

Furthermore, aromatic amino acids like phenylalanine, tryptophan, and tyrosine are crucial for creating cation-π interactions with the positively charged head groups of choline lipids [13]. Lastly, peripheral membrane proteins attach hydrophobic loops or helices to the membrane in order to interact with it. To prevent particularly deep membrane insertion that may permanently bind the protein to the membrane, these helices are often amphipathic, which means they are hydrophobic on one side and hydrophilic on the other. As a result, the protein-membrane interface's secondary structural components are also significant physicochemical descriptors [35].

1.4 State-of-the-art Tools for predicting protein-membrane interactions

Several efforts towards the design of tools that detect protein-membrane regions and lipid-binding sites have appeared in the literature [36]–[39]; however, they frequently use obsolete web connections and are mainly applied to 1D protein sequences without taking protein structural information into account [36]–[38]. To our knowledge, there are three publicly accessible methodologies for predicting protein-membrane interactions from the 3D protein structure: the Positioning of Proteins in Membrane (PPM) [40], [41], the Membrane Optimal Docking Area (MODA) [42] and the Drugging pRotein mEmbrAne Machine learning Method (DREAMM) [43], [44]. By combining an all-atom representation of a solute, an anisotropic solvent representation of the lipid bilayer, and a universal solvation model, PPM can calculate the rotational and translational positions of peripheral and transmembrane proteins within membranes [40], [41]. MODA is based on the protein-protein interface predictor PIER [42] that constructs a set of uniformly spaced spots at a distance of 5 Å from one another and from the protein surface, defining each patch as the collection of all protein surface atoms. Using atom solvent-accessible surface area (SASA) and atom type-specific weights, a score is calculated by MODA. The scores are then transferred to the surface amino acids to predict which amino acids will contact the cell membranes. DREAMM is an ensemble classifier and more specifically a voting classifier with a combination consisting of five classifiers: a linear discriminant analysis, a logistic regression, a linear support vector classifier, a decision tree classifier, and a light gradient boosting machine that was trained using experimental data and achieved an F1 score = 0.92 and an MCC = 0.84 [43], [44].

While these tools are often successful in accurately predicting which amino acids interact with the membrane, they can be time-consuming, taking several minutes to hours to predict binding sites in certain proteins. As an example, the DREAMM tool needs more than an hour to predict the membrane-penetrating amino acids of the catalytic domain of PI3Kα (p110α).

1.5 Aim of Thesis

Recent years have seen a rise in interest in the study of peripheral membrane proteins due to their significance in numerous physiological functions. In a variety of disease states, including cancer, neurological disorders, cardiovascular diseases, and infectious diseases, peripheral membrane proteins have been suggested as potential therapeutic targets[16], [18]–[24], [26]. However, the difficulty in the research of peripheral membrane proteins has hindered the discovery of new drug targets and the development of medications that target them.

Peripheral membrane proteins present several challenges for researchers, including issues related to stability, as they are not embedded in the membrane and are more prone to denaturation; complexity, as the frequent presence of numerous domains makes it difficult to define their overall structure and function; and membrane contacts, as they attach temporarily and partially [25].

Consequently, the primary goal of this thesis is to develop a reliable classifier that can predict protein-membrane interactions. As a result, the following query is answered: Can we create a model that predicts rapidly and accurately protein-membrane interfaces?

The workflow to achieve these goal includes four major milestones:

- Collect and prepare a dataset of peripheral membrane proteins with experimentally known membrane penetrating amino acids, which contain an adequate number of proteins belonging to a variety of protein families.
- Utilize pLM embeddings to train machine learning classifiers to create a less time-consuming and accurate model that predicts protein-membrane interfaces of peripheral membrane proteins.
- Utilize pLM attention maps to investigate if the information of membrane-penetrating amino acids is encoded in the pLM hidden layers.
- Train logistic regression models on the pLM attention maps to predict the protein-membrane interfaces of peripheral membrane proteins.

The overall objective of this thesis is to develop a classifier that is trained on pLM outcomes and capable of precisely predicting amino acids that penetrate the membrane while reducing the time required for the analysis.

2. METHODS

2.1 Artificial Intelligence

The term "artificial intelligence" (AI) was first coined by John McCarthy, Marvin Minsky, Nathaniel Rochester, and Claude Shannon during the Dartmouth Conference in 1956. Since then, AI has garnered significant interest and has advanced both in theory and application. Theoretical groundwork for computer science and AI was built by Turing, who created the well-known "Turing Test" to define "Machine Intelligence" and developed a number of approaches and concepts to broaden the concept of AI [45]. With the help of computers, AI has a promising future for growth, despite its modest initial development. In addition, the 21st century has seen a drastically higher trend in the expansion of the AI field, and these days, the increasing prevalence of AI is transforming our daily lives subtly and is on the verge of reshaping the globe [46], [47]. For example, some intelligent devices such as Siri or Alexa are regarded as intelligent machines with voice and thought recognition capabilities by their users [48].

AI's goal is to provide machine intelligence similar to that of humans. Achieving such a goal is thought possible because of learning algorithms that imitate how the human brain is believed to learn [49]. Machine Learning (ML) has emerged as the preferred approach in AI for developing practical software for various applications, such as robot control, speech recognition, and computer vision. Some of the primary ways that AI systems learn are through supervised, semi-supervised, and unsupervised learning techniques [49], [50]. More specifically, the most popular methods are described below, and a visual representation of the architecture is shown in Figure 8 and an example in Figure 9:

1. *Supervised Learning*: The AI system is trained on a labeled dataset where the desired output is predetermined for each input. The objective of the AI system is to understand the connection between the inputs and outputs and then apply that understanding to forecast the behavior of new, unforeseen data. In general, supervised learning can be used for classification and regression tasks. For instance, to identify the species of animal in each image, a supervised learning algorithm might be trained on a collection of tagged animal photographs.
2. *Semi-supervised*: In this case, the AI system is taught using both labeled and unlabeled data. The objective of the AI system is to infer the outputs for the unlabeled data using the labeled data to learn the relationship between inputs and outputs. When there is a dearth of labeled data, and abundance of unlabeled data, this form of learning can be helpful. This learning method aims to combine the other 2 common methods (supervised & unsupervised) where there is a lack of labeled dataset and a large amount of unknown data [51].
3. *Unsupervised Learning*: It is a training method in which an AI system is trained on a dataset without labels, with the objective of finding patterns and structures in the data without any prior knowledge of the desired output. For instance, an unsupervised learning system could be trained on a dataset of photographs and asked to find groups of related photos. Unsupervised learning can be used for clustering tasks, in which data with divergent patterns are divided into various clusters, and data with similar patterns are combined into one cluster. Additionally,

it can be used for dimensionality reduction tasks, in which data is moved from a high-dimensional space to a lower-dimensional space without sacrificing any information.

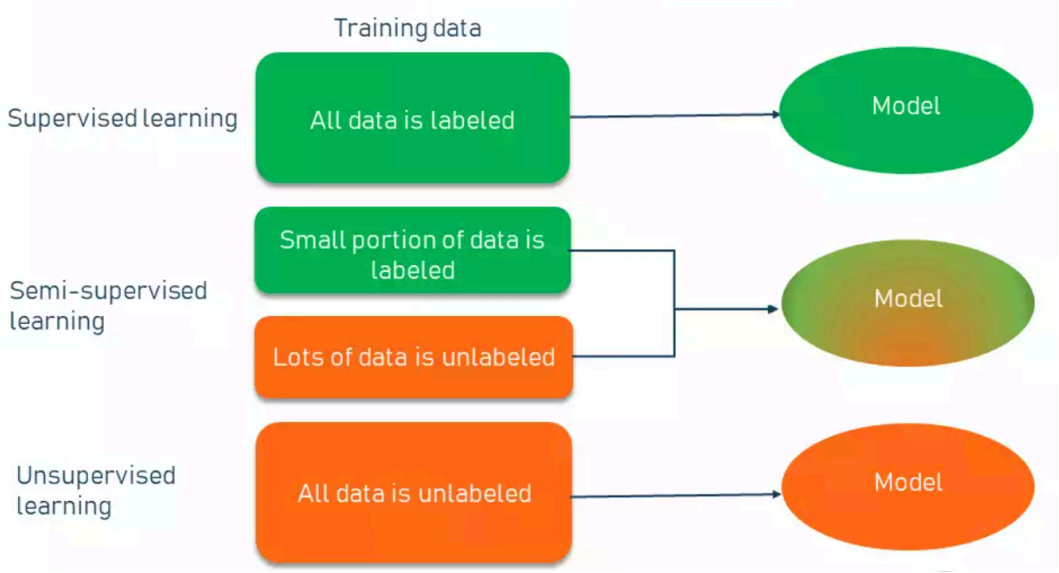


Figure 8: The 3 learning methods of Artificial Intelligence [52].

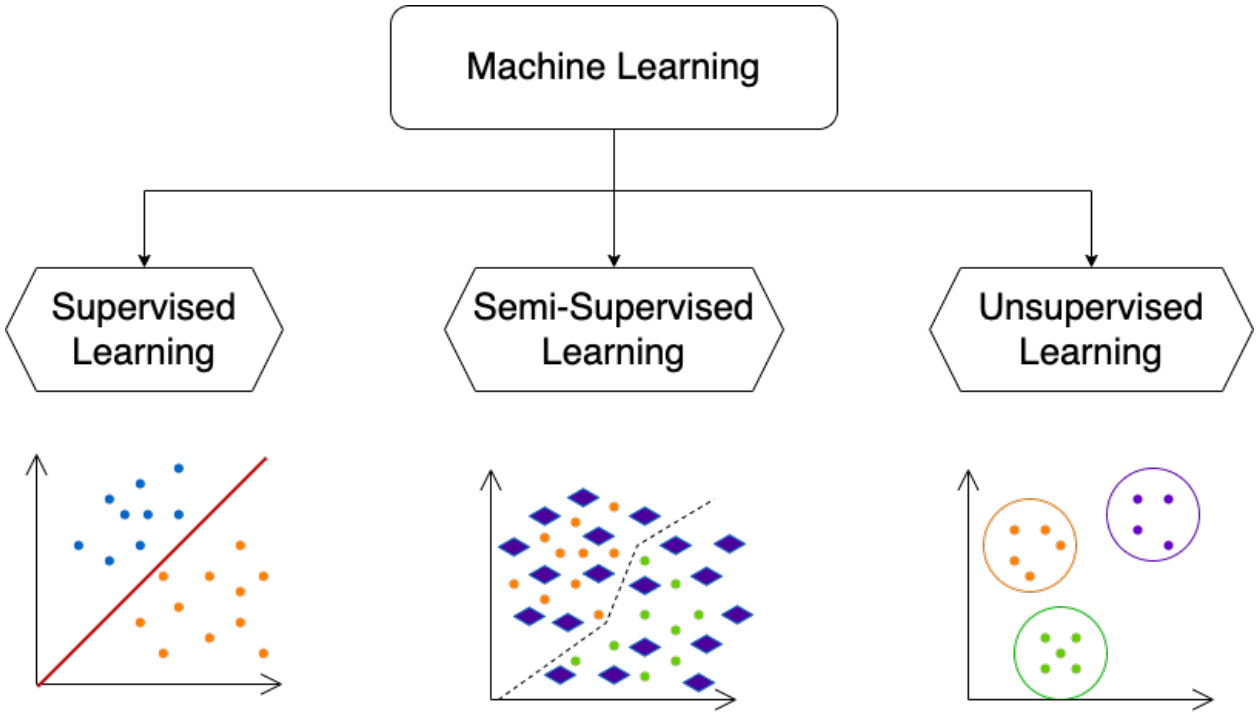


Figure 9: Machine learning methods. In supervised learning, there are labels for each sample and the model can learn to distinguish the classes – classification tasks, in semi-supervised there are few labeled samples where the model learns based on that and many unlabeled data (purple rhombus), while in unsupervised there are only samples with features that the model try to cluster them based on the characteristics – clustering task.

The optimal learning paradigm to use depends on the specific task and data available. Each of these learning paradigms has its own advantages and disadvantages. Combining

different learning paradigms and using different algorithms at different phases of the process can be applied in many real-world AI applications. Ultimately, a critical aspect of an AI system's ability to perform complex tasks and make predictions is its capability to learn from data. The objective is to endow the AI system with the ability to adapt to new, unexplored data and enhance its performance over time, whether through supervised, semi-supervised, or unsupervised learning [53].

2.2 Supervised Learning

The most widely used machine-learning methods are supervised learning methods. A function that maps an input to an output is learned through supervised learning using sample input-output pairs [53]. The objective is to generate a prediction y* in response to a query x* using a set of (x, y) pairings as the training data. The inputs x could be conventional vectors or more sophisticated data like paperwork, pictures, genetic sequences, or graphs [50].

Imagine a scenario where a model has to be developed with the eventual aim of predicting whether an individual is a patient (schizophrenia) or healthy depending on their characteristics, such as its cortical and subcortical volumes, cortical areas, thickness, etc. This can be thought of as a supervised learning problem, where the input data are the characteristics of people, and the output data are the class associated with those characteristics. The initial stage would be to compile a labeled dataset of individuals, each with a given class (healthy – not healthy). Following that, an appropriate algorithm, such as linear regression or decision trees, would be used to train the AI system on this dataset. To enable the AI system to generate precise predictions about unseen patients, the training method aims to understand the relationship between the input variables and the person's class. Once the AI system has been taught, it may be used to forecast the class of new people [54]. The AI system would produce a prediction of the category of that individual (Figure 10).

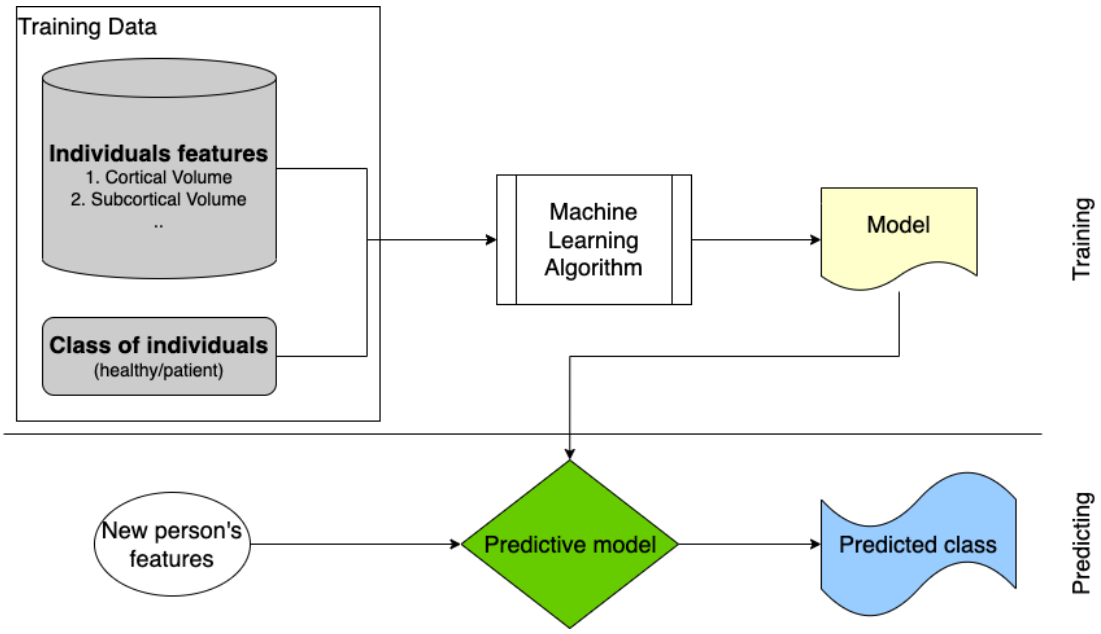


Figure 10: Machine Learning flow chart example of a model trained with people data and try to predict if a control cohort is patient or healthy based on some characteristics.

2.2.1 Dataset Generation

In machine learning, datasets play an important role as they provide the data that the AI system utilizes to learn and make predictions. Issues related to dataset creation, curation, and annotation are considered a barrier to algorithmic and scientific advancement [55]. Potential dataset issues often encountered are related to dataset quality (data obtained using different experimental methodologies, inconsistent data), quantity (not enough data), privacy and ethical considerations (medical/sensitive data), annotation (unlabeled data), and diversity. Generation and curation of high-quality datasets, suitable for use as the foundation for the training of an AI algorithm, require significant investment in time and resources [56].

The dataset is often represented in a tabular fashion. In supervised learning, a dataset usually takes the shape of a collection of labeled instances. The input data, also known as the features, and the output data, often known as the target, are both components of each example (Figure 11). Before continuing with machine learning algorithms, the data must be prepared, cleaned, and split into appropriate parts. A popular and vital machine learning technique is the division of a dataset into training, testing, and validation sets. The testing set is used to assess the AI system's performance, the training set is used to train the AI system, and the validation set is used to fine-tune the AI system's hyperparameters [57].

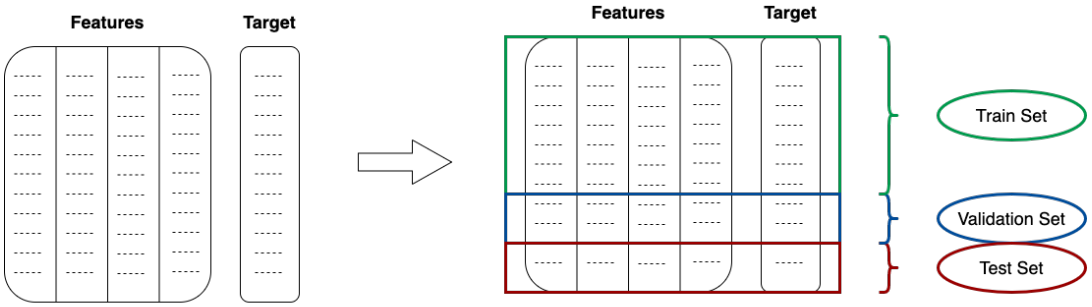


Figure 11: A table-based representation of a dataset. The dataset must be divided into training, validation, and test sets, with the former being used to train the machine learning algorithm and the latter to assess it.

The training set, which is the largest of the three sets, is used to match the AI model to the data. The AI system uses the training set to understand how inputs and outputs are related. The validation set is used to assess the system's performance to prevent overfitting. An example of overfitting is when an AI system memorizes the training set rather than understanding the fundamental connection between the inputs and outputs.

The testing set is intended to assess how well the AI system performs on fresh, unexplored data. Because it offers an unbiased assessment of the AI system's performance and helps prevent overly optimistic outcomes, using a distinct testing set is crucial.

The AI system's configuration settings, also known as hyperparameters, are tuned using the validation set. To evaluate the AI system on the testing set, the optimum hyperparameters must be identified on the validation set.

2.2.2 Feature Selection

The process of choosing the most pertinent and instructive characteristics from a dataset to utilize as inputs in a machine learning model is known as feature selection. It is a crucial stage in the preprocessing of the data and can have a big effect on how well the AI system works and how accurate it is [58]. To reduce the dimensionality of the data and remove redundant or irrelevant characteristics, feature selection aims to pick the features that are most predictive of the outcome. This can lower the computing cost, prevent overfitting, and increase the accuracy of the AI system [57].

There are several methods for feature selection, including filter methods, which rate each feature using statistical techniques like correlation or mutual information and then select the features with the highest scores, wrapper methods, that assess the significance of each feature using the AI system itself by removing characteristics from the dataset and selecting the best subset of features by an iterative procedure, and embedded methods, which combine feature selection into the AI system's training procedure by using, for example, regularization methods (Lasso, Ridge regression) to reduce irrelevant feature coefficients to zero [59].

2.2.3 Class imbalance problem

In some problems, there is a possibility of not having enough observations for a specific class (when compared to other class instances) or that a given class may not exist at all in the available data [60]. For example, a medical dataset of 1000 patients might include only 10 incidents of people that have a disease while the remaining 990 are healthy. This is known as a class imbalance issue and is found in supervised learning, where the distribution of classes in the target variable is abnormal and can happen when one class has significantly more samples than the other.

Class imbalance can pose a challenge for machine learning, as traditional techniques aim to minimize classification errors, which can be misleading when one class is underrepresented. This can result in various unfavorable effects, such as overfitting, where the model is likely to predict only the majority class, overlooking the minority class, and making incorrect classifications for new observations that belong to the minority class. Additionally, there may be algorithm bias, and inadequate evaluation of model performance, since some metrics such as accuracy, do not effectively assess the classifier's ability [60].

To handle this problem, there are several approaches [60]:

- Oversampling the minority class which is the practice of duplicating samples from the minority class to improve the balance of the class distribution.
- Taking fewer samples from the majority class in order to balance the class distribution is known as undersampling the majority class.
- Creating new synthetic samples from the minority class using methods like the Synthetic Minority Over-sampling Technique (SMOTE) [61] or Adaptive Synthetic (ADASYN) [62] sampling is known as synthetic data generation.

- Penalization of models with class weights. This method biases the model to emphasize the minority class by imposing a weighted cost when a sample is incorrectly identified. In the scikit-learn Python package [63] the weights can be automatically assigned according to Eq. 1:

$$\boldsymbol{w_j = \frac{n\_samples}{n\_classes * n\_samples_j}} \quad (1)$$

where w is the weight of class j, $n\_samples$ is the total number of samples of the j training set, $n\_classes$ is the total number of classes, and $n\_samples_j$ is the total number of samples in class j in the training set.

2.2.4 Machine learning classifiers

A machine learning classifier is used to predict the class or category of a given input sample. In other words, it assigns an input sample to one of a number of already established groupings or categories. Numerous applications, including image classification, speech recognition, natural language processing, and bioinformatics, use classifiers [64]–[66]. There are several types of machine learning classifiers, such as neural networks, support vector machines (SVMs), decision trees, random forests, linear classifiers, and k-nearest neighbors (k-NN). The specific problem and dataset, as well as the desired trade-off between accuracy, processing complexity, and other considerations, influence the classifier selection. In this work, only decision trees and neural networks were selected and analyzed.

A decision tree classifier is based on a tree-like model using a series of if/else decision rules and constantly divides the training set into subsets that maximize the separation of the data [67]. The training procedure's objective is to develop a tree structure that can correctly predict a new sample's class based on its input data. Decision tree classifiers can handle both continuous and categorical input characteristics and are easy to comprehend and interpret. However, they can be susceptible to overfitting, especially when the trees become overly complex and deep. To prevent overfitting and improve the classifier's ability to generalize, various techniques can be used, such as pruning the tree or setting a maximum depth for the tree.

As an example, we can presume that we have a dataset with fruits and each fruit has distinct color, shape, and price – features. The objective is to identify the fruit's type based on its color and shape. Starting at the root of the tree, which represents the complete dataset, we can train a decision tree classifier. The classifier checks one of the input features (such as color or price) at each internal node to divide the samples into smaller subgroups depending on the feature values. When all the samples in a subgroup belong to the same class, the process is repeated recursively for each subgroup, at which time the subgroup is represented as a leaf node (Figure 12).

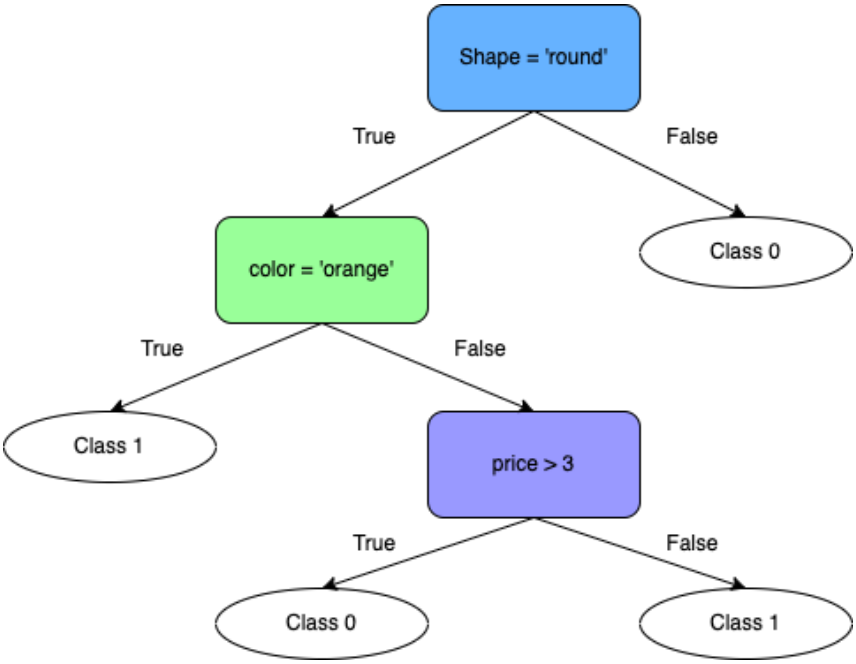


Figure 12: An example of a decision tree with three features in a binary classification task

Neural networks are also a type of machine learning classifier that go beyond traditional methods and are inspired by the structure and function of the human brain. They are made up of several interconnected processing neurons, arranged in layers (Figure 13). The input data is processed through each succeeding layer until it reaches the output layer. Each neuron in a layer takes information from the neurons in the layer below, computes it, and then sends the answer to the neurons in the layer above. The computations performed by the neurons are controlled by the weights and biases associated with each connection between them. In order for the neural network to develop the ability to generate precise predictions, these weights and biases are modified during the training phase.

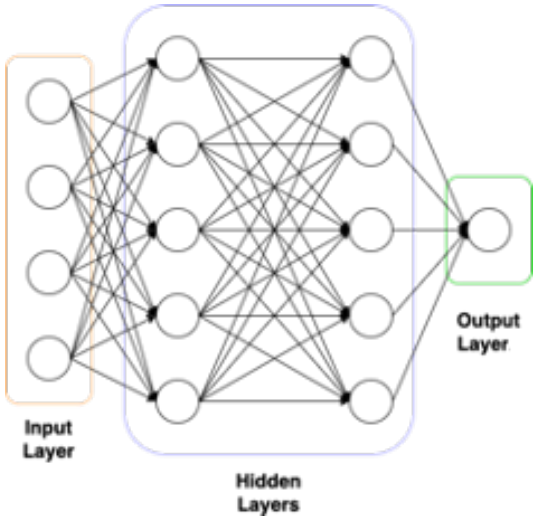


Figure 13: An example of a fully connected neural network for binary classification with one input layer, two hidden layers, and one output layer.

In particular, neural networks are effective at solving issues involving complicated, non-linear interactions between inputs and outputs. Additionally, they have the ability to recognize patterns and features in the data that may be challenging to spot using conventional techniques. Feedforward neural networks, recurrent neural networks, and convolutional neural networks are a few of the several kinds of neural network classifiers. Each kind of neural network has its own pros and cons and is created for a certain kind of challenge [68].

2.2.5 Hyper-Parameter Optimization

Each machine learning classifier has variables that affect the performance and regulate how the training algorithm behaves. These parameters, or hyperparameters, have an impact on how well the classifier can recognize patterns and correlations. Hyperparameters are variables that are predetermined rather than learned from the data during training. The performance of a machine learning model can be quite sensitive to the selected hyperparameters, making hyperparameter tuning critical. In some circumstances, a small change in the value of a hyperparameter can result in a significant change in the model's performance [69].

The most common techniques that can be used for hyperparameter optimization are:

1. Grid search involves a systematic search across a predetermined list of hyperparameter values arranged in a grid-like manner. All feasible combinations of hyperparameter values are tested and the combination that performs the best is chosen. It is an easy-to-implement, clear-cut method, but it can be computationally expensive and time-consuming if there are many hyperparameters and their potential values.
2. Random search involves searching through random combinations of hyperparameter values and is used to train instances of the model. Unlike grid search, which exhaustively tries all the given ranges, random search creates random subsets of combinations. The primary benefit of random search is that it has the potential to be more effective than grid search, particularly for big, complex models with several hyperparameters. This is because random search can focus on a smaller, randomly chosen subset rather than having to try out all possible combinations of hyperparameters [70].
3. Bayesian optimization is a technique that seeks to find an optimal set of hyperparameters by balancing exploration and exploitation using a probabilistic model based on Bayesian statistics (Eq. 2). It aims to identify the global minimum of an objective function, which represents the performance of the model on a specific task, by developing a probabilistic model of the objective function based on the observed values of the hyperparameters. Compared to grid search or random search, Bayesian optimization can be more effective, especially for complex models with multiple hyperparameters. This is because it efficiently prioritizes the search for hyperparameters likely to improve performance while also exploring other hyperparameters. By striking a balance between exploration and

exploitation, Bayesian optimization helps identify the global minimum of the objective function [71].

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)} \quad (2)$$

The following figure illustrate an example of the search procedure for the 3 methods, where the bullets represent the set of parameters that are tested in each case and the color indicates the number of trial (1st trial is the black bullet – 16th trial is the orange bullet).

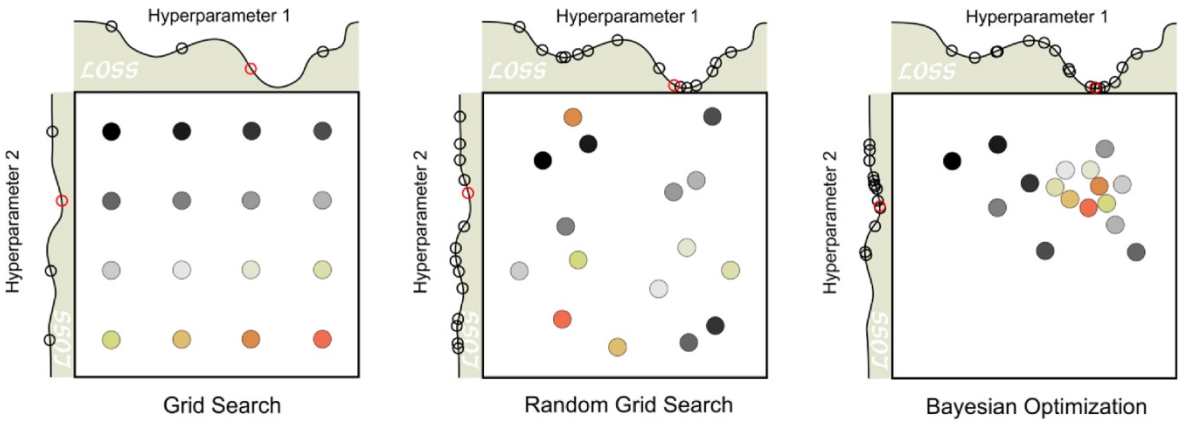


Figure 14: Hyper-parameter optimization of two parameters with sixteen search trials in the three different search methods [72]**.**

2.2.6 Performance metrics for evaluating machine learning models

To choose the most appropriate model for each problem, evaluation metrics must be used to assess the performance of different models. Depending on the task and the kind of data being examined, machine learning can employ a wide range of different assessment criteria. For example, in regression tasks, the root mean square error (RMSE) is used to evaluate the model's effectiveness. In our case, we are dealing with a binary classification problem where we need to determine whether an amino acid can penetrate the membrane or not. However, the dataset we have is imbalanced since a protein sequence contains numerous amino acids, but only a small fraction of them can interact with the membrane. Consequently, to evaluate the performance of such problems, specialized and advanced metrics are commonly used. [73], [74].

To summarize the predictions with count values for each class, the confusion matrix can be used to assess where errors in the model were made. The rows correspond to the actual classes for which the results were intended. The predictions they've made are represented by the columns.

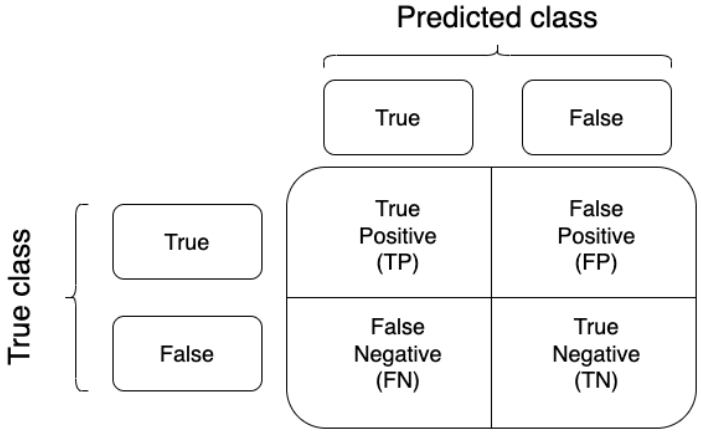


Figure 15: Confusion matrix.

Having constructed the Confusion matrix, several metrics can be calculated to quantify the performance of the model. The most common are the precision (Eq. 3) and the recall (Eq. 4) which compute what percentage of the predicted positive samples is truly positive and how good the model is at predicting the positive class, respectively.

$$precision = \frac{TP}{TP+FP} \quad (3)$$

$$recall = \frac{TP}{TP+FN} \quad (4)$$

The "harmonic mean" of sensitivity and precision is called the F-score and is commonly used in tasks with imbalanced data. F_β is a general score (Eq. 5) that uses a positive real factor β, where β is chosen such that recall is considered β times as important as precision. When recall and precision are given equal weight, the resulting score is equal to the F_1 score (Eq. 6). Values range from 0 to 1.

$$F_\beta = (1+\beta^2)\frac{precision*recall}{(\beta^2*precision)+recall} \quad (5)$$

$$F_1 = 2 * \frac{precision*recall}{precision+recall} \quad (6)$$

Another metric is the Matthews correlation coefficient (MCC) (Eq. 7). The MCC metric ranges between -1 and 1 and is formulated as:

$$MCC = \frac{TP*TN-FP*FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \quad (7)$$

2.3 Natural Language Processing

The goal of the field of study known as "Natural Language Processing" (NLP) in computer science and artificial intelligence is to make it possible for computers to comprehend, analyze, and produce human language. Creating algorithms and models that can automatically process, analyze, and understand significant volumes of natural language

data, such as text and speech, is the aim of NLP. Natural language is complicated, confusing, and context-dependent, making NLP a difficult area to study. Several broad categories can be used to classify NLP tasks, including text classification, where the goal is to identify a text document's category or mood using techniques like spam detection or sentiment analysis, speech recognition, translation, question answering, and others [75]–[77]. Simple rule-based systems and deep neural networks are two examples of complex machine learning models that can be used in NLP approaches and algorithms. These models can be used to make predictions by applying them to fresh, unforeseen data after being trained on big datasets of annotated speech and text.

NLP is playing an increasingly important role in the field of biology. The focus of this field is on processing, analyzing, and comprehending biological and medical data, including scholarly literature, electronic medical records, and clinical notes [78], [79]. In order to create structured data that can be used for a variety of applications, including drug discovery, disease diagnosis, and clinical decision support, biomedical NLP aims to automatically extract knowledge and information from massive amounts of unstructured text data, such as scientific papers and electronic medical records. The field of biomedical NLP is expanding quickly, and new innovations and uses are being created. The advancement of new medications, the comprehension of disease causes, and the enhancement of patient care are all anticipated to be significantly impacted by the usage of NLP in biology and medicine. The complexity and technicality of biological and medical data, as well as the demand for extremely high standards of correctness and dependability, are only a few of the difficulties that biomedical NLP must overcome [80]. However, the advantages of biomedical NLP make it an essential and fascinating field for study and advancement.

2.3.1 Language models

In the last years, many models have been developed to solve a variety of NLP tasks, called Language Models (LM). Most of the popular LM is based on transformer architecture. The discipline of NLP has undergone a revolution since the transformer was launched in 2017, in part because of its capacity to manage long-range relationships and parallel processing [81].

A Transformer is a type of neural network architecture that typically includes an encoder and a decoder (as shown in Figure 16). The encoder and decoder each consist of several layers of self-attention and feedforward neural networks. Self-attention is a mechanism that allows the model to selectively attend to different parts of the input sequence, generating a context-aware representation of each token. The feedforward neural networks then process these representations to derive higher-level features. Specifically, the encoder is responsible for calculating the relationship between different words of the input sequence, by attending to each token and capturing its interactions with other tokens. The last hidden layer of the encoder is commonly referred to as the “output embeddings” -or features as shown in Figure 16- which is a continuous vector representation that captures the contextual information of the input sequence. The decoder is architecturally similar to the encoder, but it adds an extra layer of masked self-

attention that only pays attention to encoded input tokens and tries to decode the output sequence of tokens. Normalization layers and residual connections are used by the encoder and decoder to stabilize training. The multi-head attention technique is similar to self-attention, but it enables the model to concentrate on many elements of the input sequence at once. This is a crucial component that gives the model state-of-the-art performance on many NLP tasks and allows it to capture long-range dependencies.

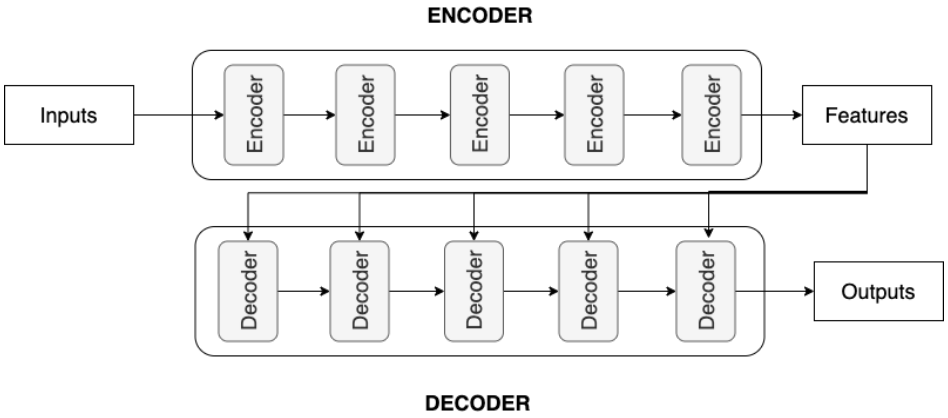


Figure 16: Transformer architecture with Encoder and Decoder part.

One of the most popular LM is BERT (Bidirectional Encoder Representations from Transformers), which was developed by Google in 2018 and is considered a state-of-the-art language representation model [82]. BERT is a transformer-based architecture model that has been trained on a large corpus of text data and can be fine-tuned for a variety of NLP tasks, such as named entity recognition, text classification, and question answering. BERT is unique and powerful because it considers both left and right contexts (bidirectional), which allows it to capture the context of words within a phrase. In contrast, conventional language models only consider either the left or right context. This feature of BERT enables it to better understand the relationships between words and ultimately improve its accuracy in NLP tasks. BERT model is consisted only of the encoder part, with 12 or 24 transformer layers, depending on the model's size (Figure 17). The input to BERT consists of a sequence of tokens that are first converted into vectors using an Embedding Layer and then passed through a series of encoder layers. During pre-training, BERT employs a masked language modeling (MLM) objective in which a predetermined proportion of the input tokens are randomly masked, and the model is taught to anticipate the original token based on the context provided by the other non-masked tokens. BERT is frequently used as a feature extractor for fine-tuning on downstream tasks, where input text is tokenized first and then run through the pre-trained BERT model to create a fixed-size vector representation for each token, that are called Embeddings.

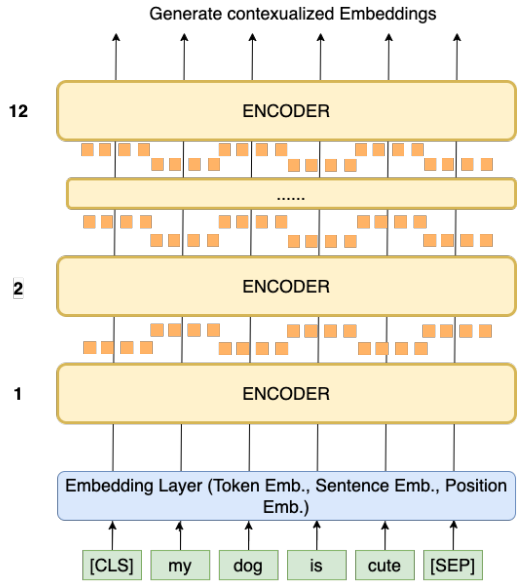


Figure 17: BERT model architecture with 12 hidden layers.

Another popular LM is T5 (Text-to-Text Transfer Transformer) which was also introduced by Google in 2019 and is transformer-based [83]. In contrast to BERT, T5 consists of both the Encoder and the Decoder part which makes it unique among transformer-based language models in that it is trained to perform a wide range of NLP tasks by treating all tasks as text-to-text problems. This means that the input and output of the model are both text strings and the model is trained to translate one string to another. T5 is extremely adaptable and, in addition to being effective on a variety of tasks, is quickly adapted to new tasks or domains by fine-tuning on a minimal quantity of task-specific data. Because of this, T5 is a preferred option for many NLP applications.

2.3.2 Attention heads

As previously explained, transformers consist of self-attention layers, which are a fundamental part of their architecture. These layers include attention heads, which allow the model to simultaneously focus on multiple positions and features of the input sequence, thereby learning diverse aspects of relationships in the data. After processing the input through all the heads, their outputs are concatenated and combined to create the final representation. Each head computes a separate attention weight distribution and is in charge of paying attention to a separate component of the input, such as the syntax of a sentence or the meaning of a certain word (Figure 18). The model's capacity to capture fine-grained dependencies and manage complex input sequences is enhanced by attention heads. The attention scores are computed independently for each attention head in a transformer model with multiple attention heads. This can increase the model's accuracy and effectiveness, especially for tasks that call for complicated or subtle interactions between various elements of the input sequence.

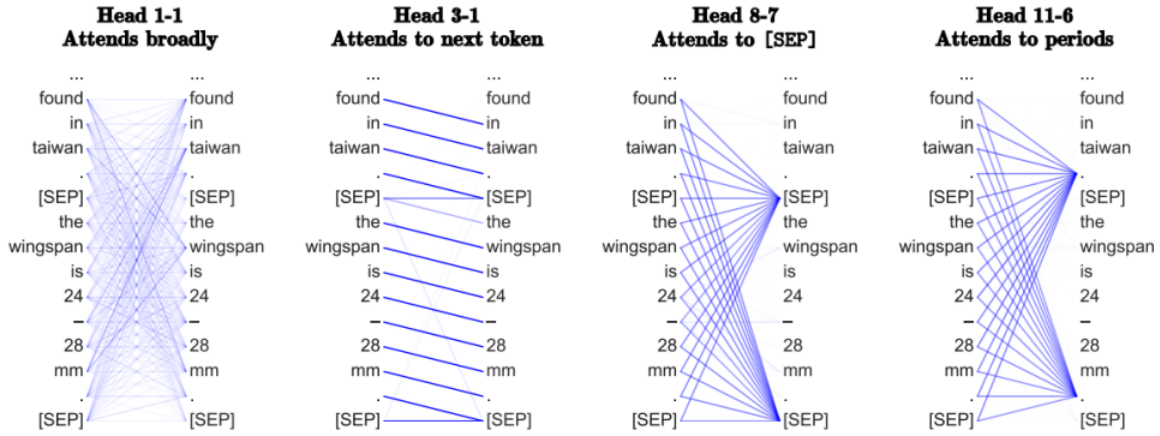


Figure 18: Example of heads paying attention to different parts. The darker lines indicate the strength of the attention weight. The darker the line, the higher the weight [99].

The Attention layer takes its input in the form of three parameters - Query, Key, and Value (Figure 19A). The Transformer repeatedly and simultaneously refers to each Attention processor as an Attention Head. Multi-head attention is the term used for this (Figure 19B). By integrating numerous similar Attention calculations, it offers its Attention a stronger capacity for discriminating [81].

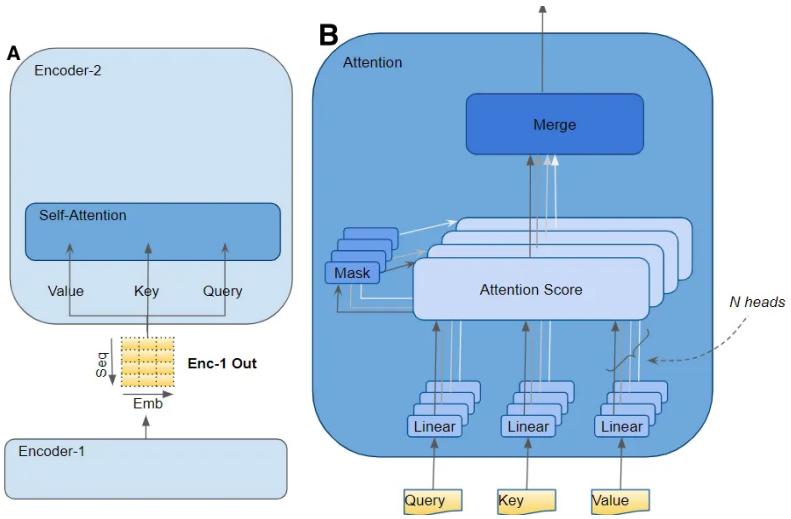


Figure 19: Input of attention layer in the form of Query, Key & Value (A). Multi-head attention (B) [100]

The following function is used to calculate the attention vectors for all tokens in a sentence:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right) \quad (8)$$

where d_k is the dimension of keys.

In biology, attention heads have been the subject of extensive research, which has revealed that different attention heads in the models' layers are responsible for specific protein characteristics, such as the secondary structure or binding site of a protein[101]. Moreover, the ESM-1b model utilizes attention maps to predict the contacts between

amino acids and it has been observed that distinct attention heads specialize in different types of contacts [87].

2.3.3 Language models in Biology

A key feature of Transformers is transfer learning, which is the application of knowledge gained from completing one task to help solve a different, but related, problem [81]. This, in combination with the fact that protein sequences are ideal for LM, has led many researchers to train new models to solve downstream tasks or to improve existing technologies [84]–[88]. Protein Language Models (pLMs) treat a protein sequence as a sentence and each amino acid as a single word, similar to NLP. For proteins, specific 3D shapes are necessary to carry out specific tasks, which impose constraints on the language and meaning, as in NLP.

Two of the most widely-known applications in the biology field are the protTrans [84] and Evolutionary Scale Modeling (ESM) [87] models. Using known databases such as Uniref50 [89], Uniref90 [89], Uniref100 [89], CATH [90], and BFD (Big Fantastic Database) [91] they collected millions of protein sequences and billions of amino acids for the model training. ProtTrans successfully trained six NLP LMs (T5 [83], Electra [92], BERT [82], Albert [93], Transformer-XL [94] and XLNet [95]) on protein sequences. The next table (Table 1) includes a part of the configurations for the pre-training of pLMs for protTrans.

Table 1: Part of protTrans configurations hyperparameters.

		Dataset	***Number of Layers***	***Embedding Dimension***	***Number of Heads***
Models	ProtXL	BFD100	32	1024	14
		Uniref100	30		16
	ProtBert	BFD100	30	1024	16
		UniRef100			
	ProtXLNet	UniRef100	30	1024	16
	ProtAlbert	UniRef100	12	1024	64
	ProtElectra	UniRef100	30	1024	16
	ProtT5-XL	BFD100	24	1024	32
		UniRef50			
	ProtT5-XXL	BFD100	24	1024	128
		UniRef50			

For ESM, researchers opted for the use of large unsupervised language models. They trained high-capacity Transformer language models on evolutionary data, and observed the model identified several key pieces of information such as homology, structural similarity, etc. In the table below, some of the main models that were developed from ESM (Table 2) are listed. It is worth noting, that ESMFold [96] achieved performance on par with that of AlphaFold [97] and RosettaFold [98], the two most known computational methods for predicting the 3D structure of proteins, while at the same time achieving a 600-fold speedup for results only marginally less accurate than those of AlphaFold2.

Table 2: Part of ESM models with its configurations.

		Dataset	***Number of Layers***	***Embedding Dimension***	***Number of Parameters***
Models	ESM-2	UniRef50	6, 12, 30, 33, 36, 48	320, 480, 640, 1280, 2560, 5120	8M, 35M, 150M, 650M, 3B, 15B
	ESMFold	UniRef50	48	-	690M
	ESM-MSA-1b	UniRef50	12	768	100M
	ESM-IF1	UniRef50 + CATH	20	512	124M

These models can be used for several tasks like secondary structure prediction, discovery of biological variations, capturing of biophysical features of amino acids, prediction of protein subcellular localization, and others. As an example, in the following picture (Figure 20) is a use case of a pLM which takes as input a protein sequence with L amino acids, which are tokenized and positional encoding is added. The resulting vectors are passed from the model that we have chosen and generate features – the process of embedding - for each input token. Then, the last hidden state of the model can be used for downstream prediction tasks, like using them as input to a Convolution Neural Network (CNN) to predict an amino acid's secondary structure or to a Feedforward Neural Network (FNN) to predict the cellular location of the given protein.

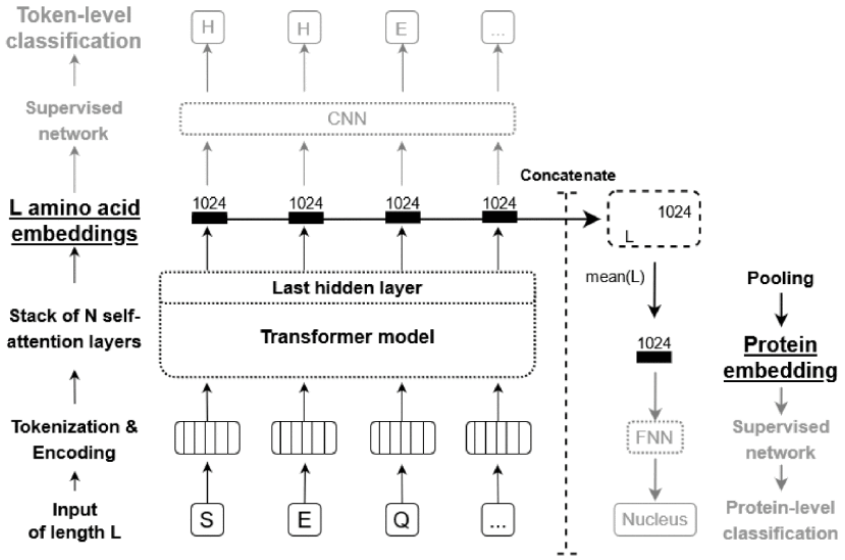


Figure 20: A general overview of how ProtTrans models can be used to derive features (embeddings) from an unknown protein sequence and used them for classification tasks [84]**.**

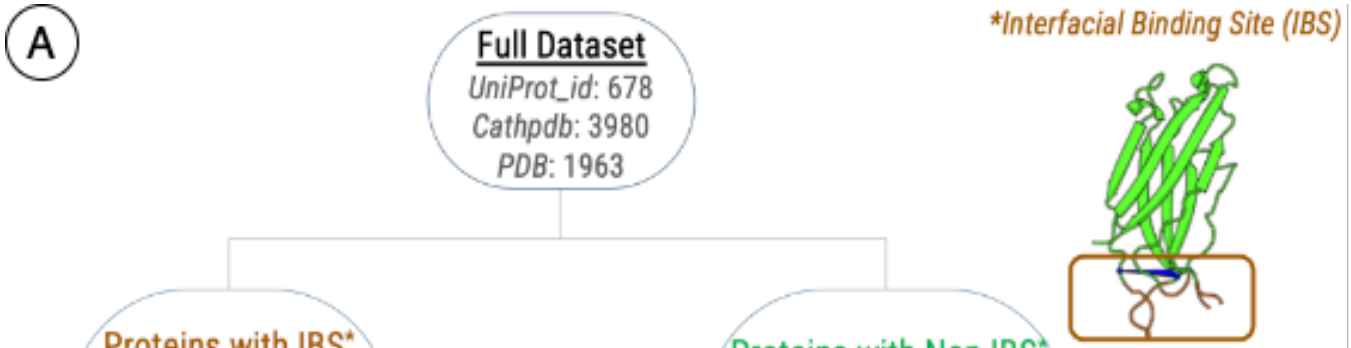
3. RESULTS

In this chapter, we analyze the procedure from data collection to the usage of machine learning classifiers and attention heads of pLMs for predicting protein-membrane interfaces. First, we describe the datasets used and how they were appropriately prepared, ensuring that desired information was retained. Next, we measured the pairwise percentage identity of the amino acid sequences between every protein in the dataset, clustered these proteins, and retained only the representative sequences of each cluster to create an unbiased dataset. With a refined dataset in hand, we proceeded to annotate each protein, and then we continued by using pLMs to construct the necessary information through the embeddings they produce as features. We then fitted the dataset into machine learning classifiers and fine-tuned the most promising algorithm. Test proteins were also used to evaluate the model's predictive ability. Finally, we analyzed the attention heads of pLMs to predict the protein-membrane interfaces.

3.1 Dataset Collection and Dataset Preparation

For dataset construction, we used two publicly available datasets that include peripheral membrane proteins. The first dataset was downloaded from “Resources for Peripheral Protein-Membrane Interactions (PePrMInt)” and includes 2.522 structures, consisting of 1.328 experimental structures from CATH [90] and 1.994 AlhaFold [97] models containing one (of nine possible) domain implicated with the membrane assocation, namely Annexin, C1, C2, discoidin C2, PH, PX, PLA, PLC/D, START [30]. This dataset was generated by first defining the membrane binding sites in each superfamily using information from the literature and then transferring that annotation to other domains in the same superfamilies, taking advantage of structural alignment. Figure 21A shows the number of proteins in the full dataset, which includes all proteins with Interfacial Binding Sites (IBS) and those without. IBS are the amino acids of a protein that interact with the membrane. Figure 21B displays a small portion of the dataset information. We noticed that one PDB ID can match multiple CATHPDB codes, as the CATHPDB code includes the PDB code and the chain ID. Also, one Uniprot ID (or “uniprot_acc” as mentioned in file) can correspond to multiple PDB IDS.

A



B

	domain	cathpdb	pdb	uniprot_acc	uniprot_id	residue_name	IBS	chain_id	residue_number
924	PH	3g9wA02	3G9W	Q71LX4	TLN2_MOUSE	ASN	True	A	326
1316	PH	3g9wB02	3G9W	Q71LX4	TLN2_MOUSE	ASN	True	B	326
4468	PH	3g9wA02	3G9W	Q71LX4	TLN2_MOUSE	SER	False	A	314
4469	PH	3g9wA02	3G9W	Q71LX4	TLN2_MOUSE	SER	False	A	339

Figure 21: PePrMInt aggregated information about proteins' id in the whole dataset, in the dataset that includes proteins with IBS and not IBS (A). An example of the data information that provided by PePrMInt with a part of columns and rows (B).

The second collection of peripheral membrane proteins was retrieved from the "Drugging pRotein mEmbrAne Machine learning Method (DREAMM)" and consists of 65 proteins with known 3D structures and experimentally known membrane-penetrating amino acids [43], [44].

From the PePrMInt dataset, only proteins with IBSs were kept and these proteins were compared and merged with the DREAMM dataset, using as the reference point the PDB ID, where they had 30 proteins in common. However, for our dataset the reference point was selected to be the Uniprot ID, so we matched each Uniprot ID with the corresponding PDB IDs & chain IDs, and for the DREAMM proteins that were not included in the IBS dataset (35 proteins) and there was no information about Uniprot ID, it was added manually (Figure 22A). Then, for a total of 709 proteins – unique Uniprot IDs, we downloaded the FASTA files from the Uniprot database [102] and matched each ID with the relative sequence (Figure 22B).

Figure 22: Part of file with the Uniprot ids to be the key and the corresponding PDBs with their chains, that we have information, to be the value (A). Part of Uniprot ids with the sequence information from Uniprot database (B). Both files are in JSON format.

3.2 Sequence Similarity and Clustering of Protein Sequences

Machine learning relies on having an unbiased dataset because it prevents the models from being trained on incomplete or skewed data, which could result in predictions and judgments that are prejudiced. Data that is typical of the entire population, as opposed to just a small subset, is said to be unbiased. This is crucial because biased data can produce biased outcomes, which can be harmful in a variety of real-world applications.

For this reason, because many of our proteins belong to the same superfamilies and as a result may have similarities in their sequences, it is important to keep only the most representative. So, we aggregate all the UniProt sequences, and using the CD-HIT [103] Suite, we found the sequences' similarities by clustering the proteins. The procedure starts by setting the longest sequence as the representative of the first cluster. Then, each remaining sequence is compared with the representative of existing clusters and if the similarity with any representative is above a given threshold, it is grouped into that cluster. Otherwise, a new cluster is defined with that sequence as the representative. In our case, we set the sequence identity cutoff to 40% and that gave us 443 clusters (Figure 23). From the clustering results, we kept only the representatives of each cluster (443 proteins).

```
"17": [
  {
    "representative": "P53810"
  },
  {
    "P16446": 99.63
  },
  {
    "Q00169": 98.89
  },
  {
    "P53812": 77.49
  }
],
"18": [
  {
    "Q62261": 60.35
  },
  {
    "representative": "O15020"
  },
  {
    "Q00963": 48.1
  }
],
```


Figure 23: Part of clustering results, showing the clusters 17 and 18 with the Uniprot IDs - the representative of each cluster and the similar Uniprot sequences with the similarity percentage in JSON format.

3.3 Protein Annotation

Because our reference point is the Uniprot ID while the information about the IBS of proteins is correlated with the PDB ID, we have to transfer this information to the Uniprot sequence. For this reason, we downloaded from the Protein Data Bank (PDB) [104] all the PDB codes that corresponds to the 443 proteins from the clustering procedure. In total, we fetched 1069 PDB proteins, which in turn they were aligned with the corresponding Uniprot sequences (Figure 24). In cases where the PDB protein is homodimer, homotrimer, etc. (chains with same sequence), only one chain was kept.


```
"P14555": [
  {
    "sequence": "MKTLLLLAVIMIFGLLQAHGNLVNFHRMIKLTTGKEAALSYGFYGCHCGVGGRGSPKDATDRCCVTHDCCYKRLEKRGCGTKFLSYKFSNSGSRITCAKQDSCRSQLCECDKAAATCFARNKTTYNKKYQYYSNKHCRGSTPRC"
  },
  {
    "1AYP_A": [
      {
        "seq": "--------------------NLVNFHRMIKLTTGKEAALSYGFYGCHCGVGGRGSPKDATDRCCVTHDCCYKRLEKRGCGTKFLSYKFSNSGSRITCAKQDSCRSQLCECDKAAATCFARNKTTYNKKYQYYSNKHCRGSTPRC"
      },
      {
        "uni_seq": "MKTLLLLAVIMIFGLLQAHGNLVNFHRMIKLTTGKEAALSYGFYGCHCGVGGRGSPKDATDRCCVTHDCCYKRLEKRGCGTKFLSYKFSNSGSRITCAKQDSCRSQLCECDKAAATCFARNKTTYNKKYQYYSNKHCRGSTPRC"
      },
      {
        "res_num": [
          "ASN1",
          "LEU2",
          "VAL3",
          "ASN4",
```


Figure 24: Part of P14555 protein and the alignment with the 1AYP chain A PDB protein in JSON format.

Having aligned all the sequences, we set the numbering of residues based on the UniProt sequence. Then, we update the IBS amino acids of Uniprot sequences, based on the annotation of each corresponding aligned PDB sequence. That was the last step for the proteins' preparation and annotation (Figure 25A). As it was expected, the dataset is totally imbalanced with only 3% of the total number of amino acids to belong to the IBS class (Figure 25B).

A

id	uniprot_id	residue_1l	residue_3l	residue_index	is_IBS	sequence
275	P80966	C	CYS	132	0	MNPAHLLVLSAVCVSLLGASSIPPQPLHLIQFGNMIQCTVPGFLSW...
276	P80966	F	PHE	133	0	MNPAHLLVLSAVCVSLLGASSIPPQPLHLIQFGNMIQCTVPGFLSW...
277	P80966	A	ALA	134	0	MNPAHLLVLSAVCVSLLGASSIPPQPLHLIQFGNMIQCTVPGFLSW...
278	P80966	A	ALA	135	0	MNPAHLLVLSAVCVSLLGASSIPPQPLHLIQFGNMIQCTVPGFLSW...
279	P80966	S	SER	136	0	MNPAHLLVLSAVCVSLLGASSIPPQPLHLIQFGNMIQCTVPGFLSW...
280	P80966	P	PRO	137	0	MNPAHLLVLSAVCVSLLGASSIPPQPLHLIQFGNMIQCTVPGFLSW...
281	P80966	Y	TYR	138	1	MNPAHLLVLSAVCVSLLGASSIPPQPLHLIQFGNMIQCTVPGFLSW...
282	P80966	N	ASN	139	1	MNPAHLLVLSAVCVSLLGASSIPPQPLHLIQFGNMIQCTVPGFLSW...
283	P80966	N	ASN	140	1	MNPAHLLVLSAVCVSLLGASSIPPQPLHLIQFGNMIQCTVPGFLSW...
284	P80966	N	ASN	141	1	MNPAHLLVLSAVCVSLLGASSIPPQPLHLIQFGNMIQCTVPGFLSW...

B

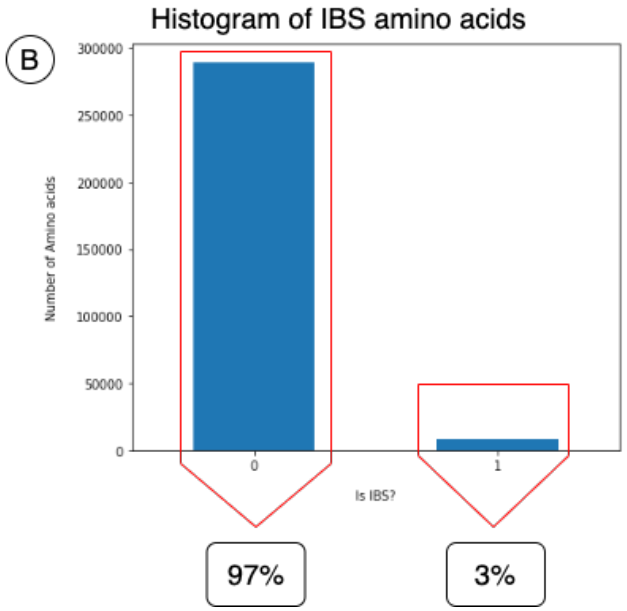


Figure 25: Part of the dataset depicted the information of Uniprot ID, the residue name (1 & 3 letter code) and its index, if a residue is IBS, and the Uniprot sequence (A). The total number of amino acids that belongs to each class (B).

To sum up, the next figure concentrates the process for dataset preparation, filtering and annotation that was described in the last 3 sections.

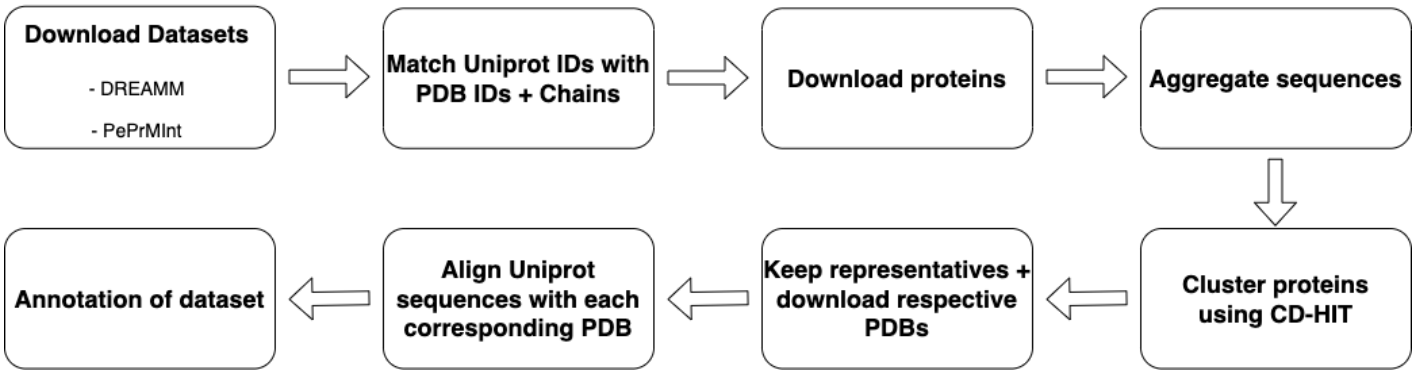


Figure 26: Pipeline for dataset preparation.

3.4 Construction of Protein Embeddings

To generate the features of our dataset, we used two pLMs. These models produce embeddings that represent amino acids or groups of amino acids in a fixed-dimensional vector space. These vectors encode the structural and functional properties of a protein and serve as ideal features for making predictions about a protein's structure, function, and properties based on its amino acid sequence. We employed the *prot_t5_xl_half_uniref50-enc*[1] model from protTrans [84] and the *esm2_t33_650M_UR50D*[2] from ESM [96], both of which are available in the Hugging Face[3] repository. The former produced embeddings vectors with a size of $sequence\_length \times 1024$ while the latter yields embeddings of size $sequence\_length \times 1280$ (Figure 27). The procedure for extracting this information was highly efficient, taking only for a medium-length protein (up to 512 amino acids) ~10 seconds for protTrans and ~3 seconds for ESM in a MacBook Pro with chip M1, 10-cores, and 16GB of RAM.

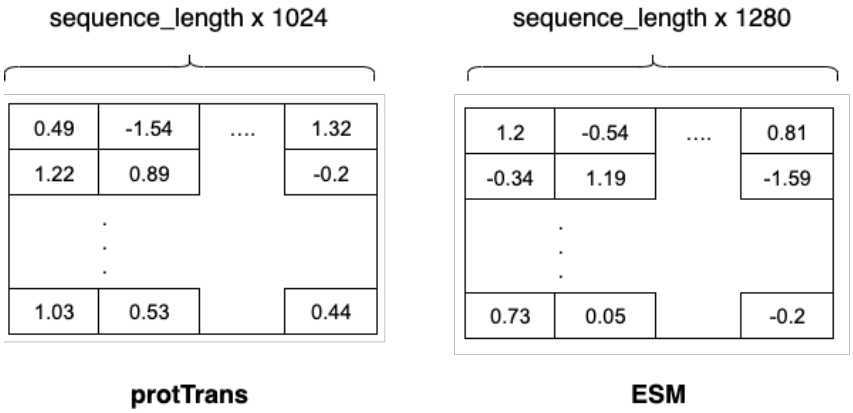


Figure 27: Embeddings dimensions from protTrans and ESM pLMs.

Having generated the embeddings, we added this information to our dataset as features (1 dataset for protTrans embeddings & 1 dataset for ESM embeddings). The last step that was performed for features was to convert the amino acid feature column from categorical to numerical features, by using one-hot encoding, a technique that transforms each unique value in a categorical feature into a new binary feature (Figure 28).

[1] https://huggingface.co/Rostlab/prot_t5_xl_half_uniref50-enc

[2] https://huggingface.co/facebook/esm2_t33_650M_UR50D

[3] https://huggingface.co/

id	uniprot_id	residue_1l	is_IBS	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
0	P14555	M	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
1	P14555	K	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
2	P14555	T	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
3	P14555	L	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
4	P14555	L	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
5	P14555	L	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
6	P14555	L	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
7	P14555	A	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 28: Part of the dataset with the one-hot encoding of the residues.

3.5 Predicting Protein-Membrane Interactions of Peripheral Membrane Proteins Using Artificial Intelligence

To calculate the performance of the algorithms, we need to test them in unknown data and record their performance. For that reason and before moving on, we split our dataset into a training set (~80% of the dataset - 328 proteins), validation set (~10% - 41 proteins), and test set (~10% - 41 proteins) (Table 3). Furthermore, an extra test set of 35 peripheral membrane proteins with known 3D structures and experimentally known membrane-penetrating regions (not amino acids) was created. These proteins were evaluated qualitatively because no particular amino acids were examined in experiments.

Table 3: The peripheral membrane proteins of the training, validation, and test sets with the corresponding Uniprot IDs.

Set	Uniprot IDs
Training	O01761, Q9JKS6, Q60841, O75962, P21359, Q12802, P26039, V5M2P5, O15020, P00451, A0A0H2UP19, P12259, P97479, P32639, O75643, Q8XM24, O75923, Q9BZ29, D3ZJP6, Q9ERC5, Q3T552, A0A0H2YST8, Q5T5U3, D4QAP3, Q6ZPE2, Q14185, Q8DR60, Q04205, E5RWQ2, Q62768, Q15811, Q6ZPF3, Q8WZ64, Q61194, A0A0H2UNT5, Q8TCU6, P26831, Q9JIR4, P71140, Q13009, Q9JIS1, Q9NZN5, O15085, Q9Y2I1, Q6ZNL6, Q7SZN0, O75747, Q63HR2, Q69ZL1, Q62868, Q6EDY6, Q9ULU8, Q07889, P46934, Q96L93, Q68CZ1, P10686, Q96KN7, Q70E73, Q45712, P35568, Q01970, Q16760, P49796, Q99490, Q00722, P16480, P05068, Q08236, Q8XMY5, B3LEP7, Q64096, Q9QWY8, F2YQ19, Q9NQW6, Q0PRN1, Q9HAU0, Q8IX03, Q9Y3M8, P48736, P14090, Q9ZA17, Q8Y4J2, Q9Y2J2, O69230, P51584, P42337, Q54873, Q91VS8, Q8BTI9, P29323, Q00944, Q9Y2H5, Q9Y5B9, A0A0H2US34, P32558, Q56F26, Q82PP4, Q8VNN2, P00723, Q8A2X6, Q70SY0, Q8KRF6, Q9KG76, Q8IWE5, Q0TR53, Q6P4T1, Q7WTN6, Q8N960, Q92974, Q9VFS5, P94286, Q6DN90, Q93RE7, A3DHG6, Q9Y5W7, A0A075B5H6, O60462, Q59290, Q9BYX2, A0FGR8, Q08345, Q92888, P77847, Q96J02, P0C2S1, O94806, Q6ZUM4, Q9BZF1, Q6DN12, Q61097, P50570, P11171, Q8AAK6, Q16832, D1GCC6, P15498, A3DK57, Q840C0,

	Q8N4X5, A6KXE5, P10477, O95267, Q93IE7, Q8AB22, Q9H4M7, Q8K4I3, P22346, Q9HCE7, P10688, Q9BRR9, P47712, Q9BXB4, B3CET4, Q15027, P09216, O00522, Q99PV3, A4I5U9, Q8A916, Q92556, Q3J126, Q5JSP0, Q9UH99, O54924, Q8R5F8, Q08945, Q02111, Q5LJ68, P05129, P9WG63, Q5LIC7, Q9NR80, P21146, P40485, P47709, O52780, P0CS93, Q96AC1, Q9RIK9, P16278, P79134, Q96C24, Q8VRK8, Q13613, Q2PHL4, Q3ZC95, Q82L26, P08236, Q4AE70, P04049, Q02834, P42331, Q9ZB22, P32776, Q9WXN1, P45796, P0A377, P18887, Q9H0H5, Q9L5A4, Q9Y5P4, O07653, Q9JID9, Q9Y217, Q8WXI4, Q59675, Q9X0S8, P05804, P46662, A0A0H2UN19, Q9Y5X1, Q13322, Q6DN99, P40748, A2AR50, Q9UH65, Q51815, Q9NPI6, Q8WV41, Q9L9D7, G0SHK5, Q06696, H6WCZ0, A0A0H2WZL3, D8DVU6, O15530, P10820, P29366, P32780, Q96L92, Q00019, Q91X46, Q13596, Q9BPZ7, Q8WU20, Q18PE0, Q61234, Q9ERE3, P16559, Q4W8M3, Q8A5P6, G0L2L9, P97465, P31751, B7GNN8, Q9BSW7, Q15036, P52757, P21956, Q8AAM3, Q8A3J5, P40161, Q38CF2, Q14849, Q9BSQ5, Q8A9F0, Q7L8C5, Q9ERS5, Q9H2B2, P21579, Q5EBH1, O60496, B1H267, O43581, O43739, P14598, Q9H0F6, Q9UNH6, Q86VN1, E0RVY7, Q8NFA2, O75689, Q9W1H5, A7LSX5, C4QH88, D6MSV6, Q96MF2, B3PDE5, Q86WV1, B3PIB0, P09394, Q9NYT0, P08567, B3A043, P25335, Q5L9W9, P04272, Q15080, O95433, Q5LFR2, Q9SSK9, A0A0J9X278, P08954, P34024, Q9SYT0, A6LIT8, P32912, A0A6L7H2E6, P45723, Q5LX22, Q2LK81, Q8I914, D7RFJ9, Q2PA00, Q4VPP2, Q9P104, Q8C4Q6, Q8A2Z3, Q9HB20, Q80UW2, P17063, Q9ULZ2, P49675, Q9UN19, Q9CR95, A0A6N4SPL7, P53810, Q9Y5W9, Q9RZE3, A6L916, P42530, P53068, A9CLR1, A9CG82, Q5JGZ3, A0A0H2V2B5, Q9QYE9
Validation	Q8H1L1, Q1MFM4, Q9I4D2, G7J032, P52778, Q3J4M4, Q5HLI9, Q8NN40, O04298, A1U5H9, Q9XG81, Q2K6S8, Q2Y8N9, Q5QL47, P80966, Q6UV28, Q47KK8, Q9A7I7, Q5SK03, Q7NY36, Q9C8S6, Q9Y9R3, A0A6N4SU23, Q484T9, O31806, P14555, Q5LN61, Q9Y547, Q98IT8, Q689C4, Q81AY6, Q8DVN6, A0A6N4SXV3, A0KKT0, Q973T5, Q49US3, Q7CZ16, E2FYL5, Q64YT5, Q9P805, P0C0B0
Test	P04183, Q89ZG6, Q12517, Q53W25, Q99JV5, Q9PPP5, Q8VZS8, P59095, A1ZAW5, Q9UKL6, Q9NSY2, Q9P4F6, Q67A25, Q8PPZ5, Q9HJ63, O14713, Q96L94, B8LIX8, Q9UM13, A0A1C9V3S9, Q9WYN2, Q7WAN9, Q8KNE9, Q8KNF0, B9PKK4, Q82XK1, Q8PZJ2, Q9F6D3, A0A0H2XIZ7, A0A6N4SQ07, P00630, Q9ZLJ5, A1RA60, O15496, B9PJE6, P93330, Q08826, Q9UMY4, Q98FZ2, Q832L1, A1JSS7
Extra Validation	P61914, Q15075, O16025, Q96QK1, Q960X8, P12530, P00735, P40343, O24592, P05979, O88339, P22637, Q28175, P08684, P0C2E9, Q9LCB2, P60484, P0C216, Q02127, P20932, P02749, P11889, Q77DJ6, P00803, Q99685, P49638, Q9NZD2, P00720, P12724, P12104, P56254, P60980, P01441

3.5.1 Usage of machine learning classifiers

Four machine learning classifiers were trained on both datasets (ESM & protTrans) for the training set: XGBoost classifier [105], BalancedRandomForest classifier [106], and two Neural Networks (NN) from the keras library [107]. The classifiers were trained using default parameters except for the parameter that handles the imbalance issue of the dataset. For XGBoost, we experimented with the *scale_pos_weight* parameter, while for NNs, we tested the *class_weight* parameter. The performance of each classifier was evaluated on the test proteins, and Tables 4 and 5 summarize the results.

Table 4: Models performance on protTrans test dataset with default parameters except for the class weight. The best score for each metric is highlighted in bold.

	XGBoost	BalancedRandomForest	1-layer Perceptron	Multi-Layer Perceptron
F_1 score	0.581	0.497	0.582	**0.627**
MCC	0.522	0.441	0.549	**0.586**
Confusion matrix	no_IBS IBS no_IBS 11180 941 IBS 573 1049	no_IBS IBS no_IBS 10000 2121 IBS 385 1237	no_IBS IBS no_IBS 10081 2040 IBS 453 1169	no_IBS IBS no_IBS 11702 419 IBS 689 933
Total fit time[*]	~10m	~1-3m	~1m30s	~1m

Table 5: Models performance on ESM test dataset with default parameters except for the class weight. The best score for each metric is highlighted in bold.

	XGBoost	BalancedRandomForest	1-layer Perceptron	Multi-Layer Perceptron
F_1 score	0.418	0.406	0.496	**0.553**
MCC	0.365	0.338	0.425	**0.492**
Confusion matrix	no_IBS IBS no_IBS 11595 526 IBS 1055 567	no_IBS IBS no_IBS 986 2101 IBS 395 1200	no_IBS IBS no_IBS 10032 2089 IBS 432 1190	no_IBS IBS no_IBS 11314 807 IBS 693 929
Total fit time[*]	~12m	~3m	~1m30s	~1m

[*] In a MacBook Pro with chip M1, 10-cores and 16GB of RAM.

The most promising classifier in both cases was the Multi-Layer Perceptron (MLP), which achieved the best scores and a balance between false positives (FP) and false negatives (FN), with a reasonable fitting time.

3.6 Optimization of promising algorithms

Having chosen the MLP as the algorithm with the best performance, we continued by optimizing the classifier in order to discover the best hyper-parameters that separate best the two classes. A Bayesian optimization using the *keras_tuner* library from keras [107] was performed in a wide range of values for each hyper-parameter set (Table 5) for 50 iterations. The best hyper-parameter sets based on the Bayesian search were selected (Table 5) and the respective models were saved. The present study involved performing model fine-tuning on two distinct datasets, namely the ESM dataset and the protTrans dataset. The multilayer perceptron (MLP) model was subjected to fine-tuning on both datasets. Additionally, Bayesian optimization was utilized to fine-tune XGBoost classifier specifically on the protTrans dataset, although no significant improvement was observed in comparison to the untuned model and no further investigation was done.

Table 6: The hyper-parameters that were sampled, the ranges that were searched in Bayesian optimization, and the final best hyper-parameters for each dataset using weights.

Dataset	Hyper-parameter ranges selected	Best hyper-parameters identified by hyper-parameter optimization
protTrans	*Initial Dense Layer*: 50 ≤ int(nodes) ≤ 350, step = 50 *Initial Dense Dropout percentage*: 0.0 ≤ float(dropout) ≤ 0.8, step = 0.1 *Number of Dense Layers*: 1 ≤ int(layers) ≤ 4, step = 1 *Hidden Dense Layers*: 32 ≤ int(nodes) ≤ 512, step = 32 *Dropout percentage*: 0.0 ≤ float(dropout)≤ 0.5, step = 0.1 *Optimizer*: [‘Adam’, ‘SGD’] *Learning rate*: [0.1, 0.01, 0.001, 0.0001]	*Initial Dense Layer*: 350 *Initial Dropout percentage*: 0.8 *Number of Dense Layers (Hidden)*: 4 *Hidden Dense Layers*: [512, 32, 32, 480] *Dropout percentage*: [0, 0, 0.5, 0.3] *Optimizer*: ‘Adam’ *Learning rate*: 0.1 *Class weight*: {0:1, 1:34}
ESM		*Initial Dense Layer*: 500 *Initial Dropout percentage*: 0.8 *Number of Dense Layers (Hidden)*: 4 *Hidden Dense Layers*: [32, 32, 32, 512] *Dropout percentage*: [0, 0.5, 0.5, 0.2] *Optimizer*: ‘Adam’ *Learning rate*: 0.0001 *Class weight*: {0:1, 1:34}

In Figure 29, the architecture diagrams of the two MLPs are presented with each layer, the number of corresponding nodes, and the activation functions for each Dense layer.

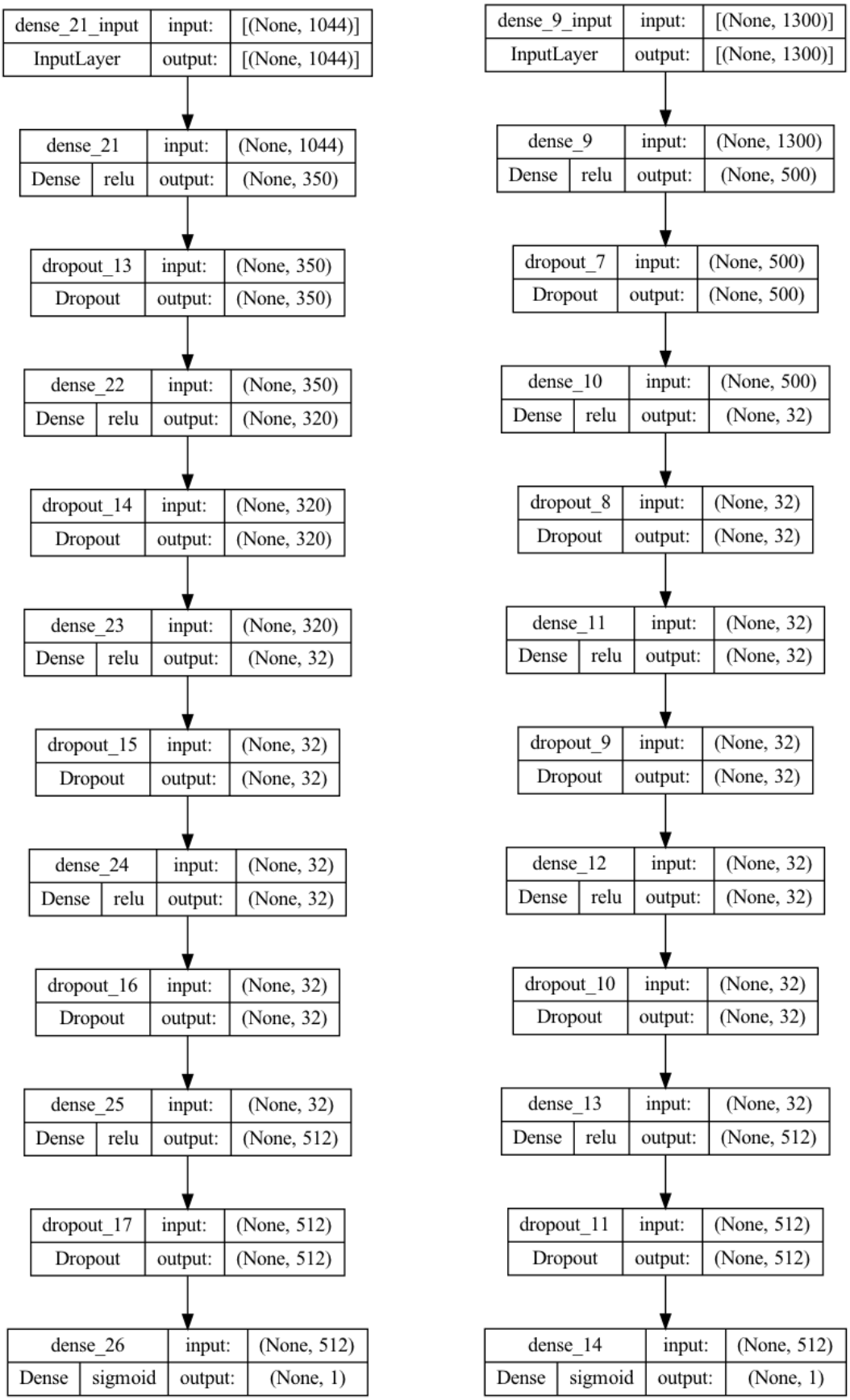


Figure 29: Architecture of MLPs models that were constructed based on the best hyper-parameters. Left is the protTrans model - Right is the ESM model.

3.7 The Test Set Prediction Results

After identifying the best hyper-parameters for the two MLP models, we proceeded to evaluate their performance on the test set. The models generated predictions in the form of probabilities, so we used the F_1 score to determine the appropriate threshold for classifying the target column. The threshold that maximized the F_1 score was selected for each model. We then evaluated the models' performance using various scoring metrics, which are presented in Table 7. Based on the prediction results, the models exhibited an improvement of approximately 7% in the metrics compared to the default classifiers.

Table 7: The predictions of the test set from the MLP models in the datasets with the protTrans and the ESM features.

Dataset	Threshold	F_1 score	MCC	TN	FP	FN	TP
protTrans	58%	0.691	0.652	6591	321	238	626
ESM	55%	0.622	0.577	6623	289	343	521

To better understand the predictions of the models, it is essential to visualize the results of the models and inspect which amino acids are FP, FN, and TP. To do this, we used the PyMOL[4] tool, which is an open-source molecular visualization system. In Figure 30 there are captures of 4 proteins' predictions from the test set using the 2 different datasets and models. The TP amino acids are depicted in green, while the FP and FN amino acids are in yellow and red, respectively (protein ligands are indicated in purple). As we can see, approximately two-thirds of the FPs are, in fact, correct predictions, as they are located in the protein-membrane interface adjacent to the true positives or on adjacent loops.

[4] https://pymol.org/2/

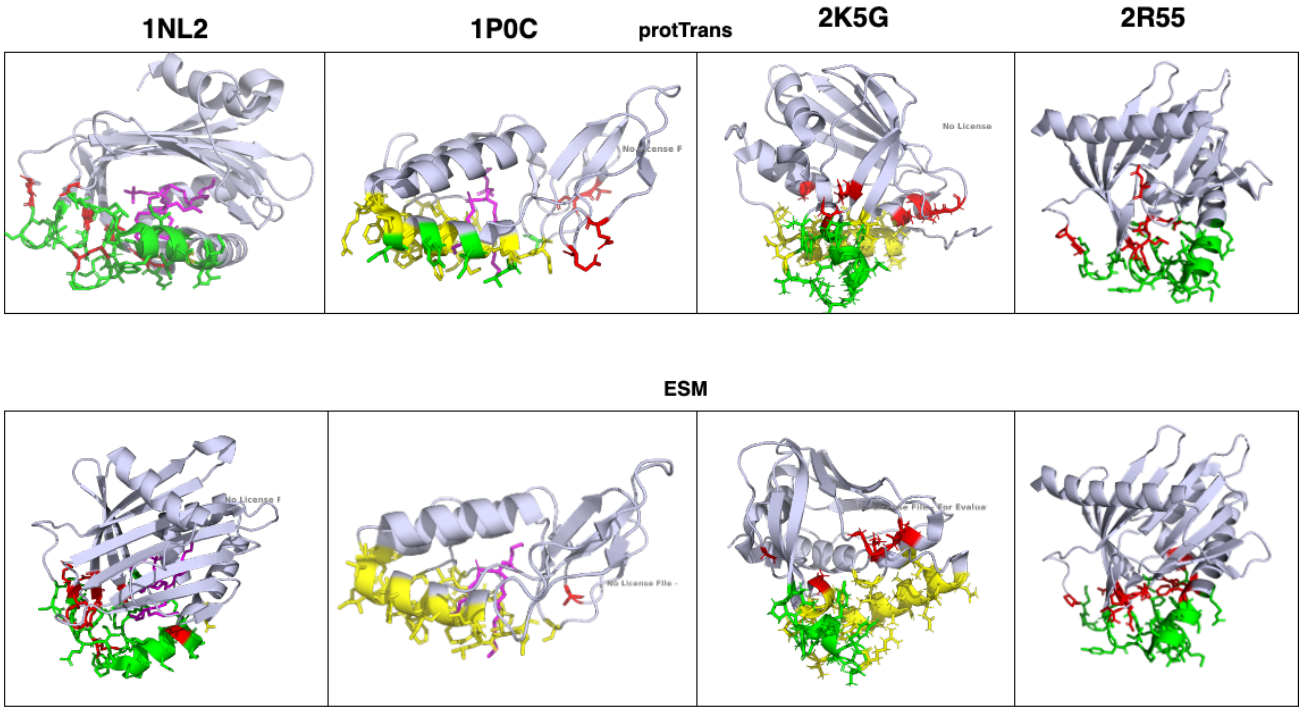


Figure 30: Four proteins of the test set visualized in the PyMOL tool with the relative predictions from the models.

For these four test proteins from the above visualization, we calculated the F_1 score independently for each protein, before and after considering that some FPs can actually be TPs due to their proximity to the residues that were originally labeled as TP based on the literature findings. In order to reclassify an FP amino acid as a TP, it was necessary for it to be located in close proximity to true membrane-interacting amino acids, and in particular, within 3Å distance of a true positive value. The resulting scores are presented in Table 8, which revealed that the actual scores are significantly higher, with an improvement of +11%.

Table 8: Calculated F1_score of the 4 test proteins before and after of converting the close amino acids that are indeed membrane penetrating, from FP to TP.

	protTrans	ESM	
F_1 score	0.87	0.84	*1NL2*
Actual F_1 score	0.98	0.97	
F_1 score	0.27	0	*1P0C*
Actual F_1 score	0.36	0.06	
F_1 score	0.52	0.42	*2KG5*
Actual F_1 score	0.67	0.5	

F_1 score	0.87	0.82	*2R55*
Actual F_1 score	0.98	0.98	

Furthermore, due to our observation that the MLP model trained on protTrans features exhibited superior performance compared to the MLP model trained on the ESM features, we chose to contrast it with the DREAMM tool. This comparison was conducted on the four proteins in the test set (1NL2, 1POC, 2K5G, 2R55), and we generated visualizations that not only highlight the membrane-penetrating amino acids but also document and report the prediction times of both models. Figure 31 illustrates the performance of the models, from which we can conclude that the DREAMM tool requires significantly more time compared to the MLP model, with its prediction time being dependent on the sequence length. However, the DREAMM tool performs equally well in prediction accuracy when compared to our model (Table 9).

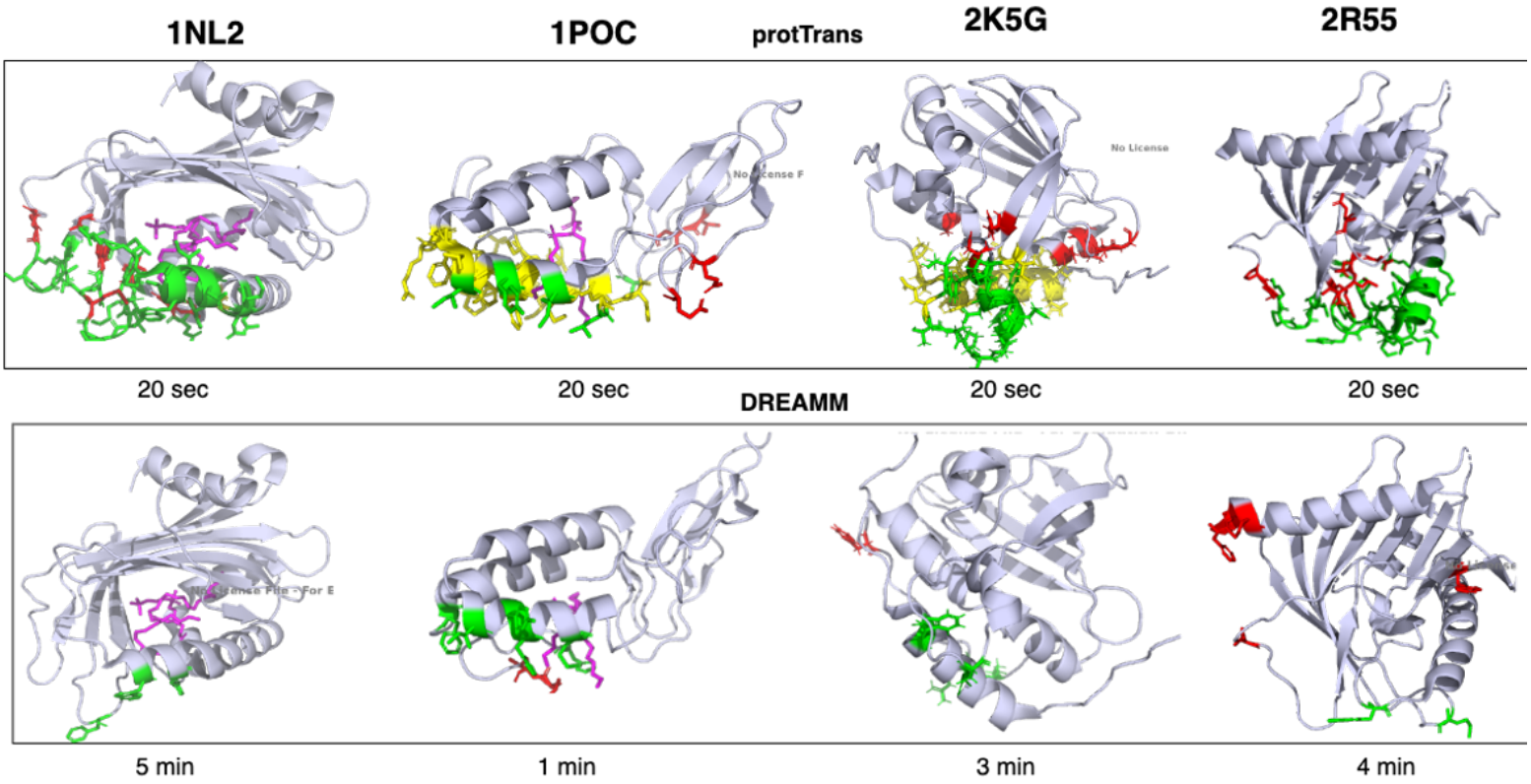


Figure 31: Comparison of four proteins of the test set between the MLP model trained on protTrans features and the DREAMM tool.

Table 9: Predicted membrane-penetrating amino acid indexes of MLP model trained on protTrans features and DREAMM tool.

	MLP trained on protTrans features	DREAMM
1NL2	83, 84* ,86, 103*, 104, 105, 106, 107, 108, 109*, 110, 111, 112*, 144*, 145*, 146, 147, 148, 149, 150, 180*, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 193, 194, 198	107, 186, 190
1POC	51*, 52*, 55, 78, 81, 85, 90, 107*, 110*	1*, 2*, 11, 24, 78, 81, 82, 86
2K5G	64*, 66*, 67, 68, 71, 72, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 109*, 110*, 140*, 141*, 144*, 154*	8, 15, 16, 78, 169
2R55	45*, 81, 82*, 84, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114*, 143*, 144*, 145, 146, 147, 148, 149, 179*, 180*, 181*, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 195	2*, 3*, 5, 40, 108, 141, 183
	**False positives residue indexes*	

3.8 Model Limitations

Next, we evaluate the performance of our models in the extra validation set, for which we do not have the exact membrane-penetrating amino acids; only the region(s) that interact with the membrane has been verified experimentally. This test shows that for protein families for which the models have not been trained on, there are no amino acids predicted at the IBS. For the ~50% of the proteins in this set, for which the model have have been trained on, the models managed to correctly predict the protein-membrane interface (Figure 32 – Table 10).

In addition, the MLP model that was trained on the protTrans embeddings and achieves better scores compared to the MLP model trained on ESM embeddings, gives approximately 20% more predictions. Finally, the MLP classifiers were also assessed for

the prediction of the protein-membrane interface of some transmembrane enzymes (Uniprot IDs: Q55487, P08842, Q99T05, P02919, P13516, O00767, Q03529, B9KDD4, O29867), but without success, as they cannot predict any membrane penetrating amino acid. In summary, the two models are limited to the specific protein families (Figure 33) that they have been trained on as was demonstrated by the results of the extra validation set, which includes unknown protein families.

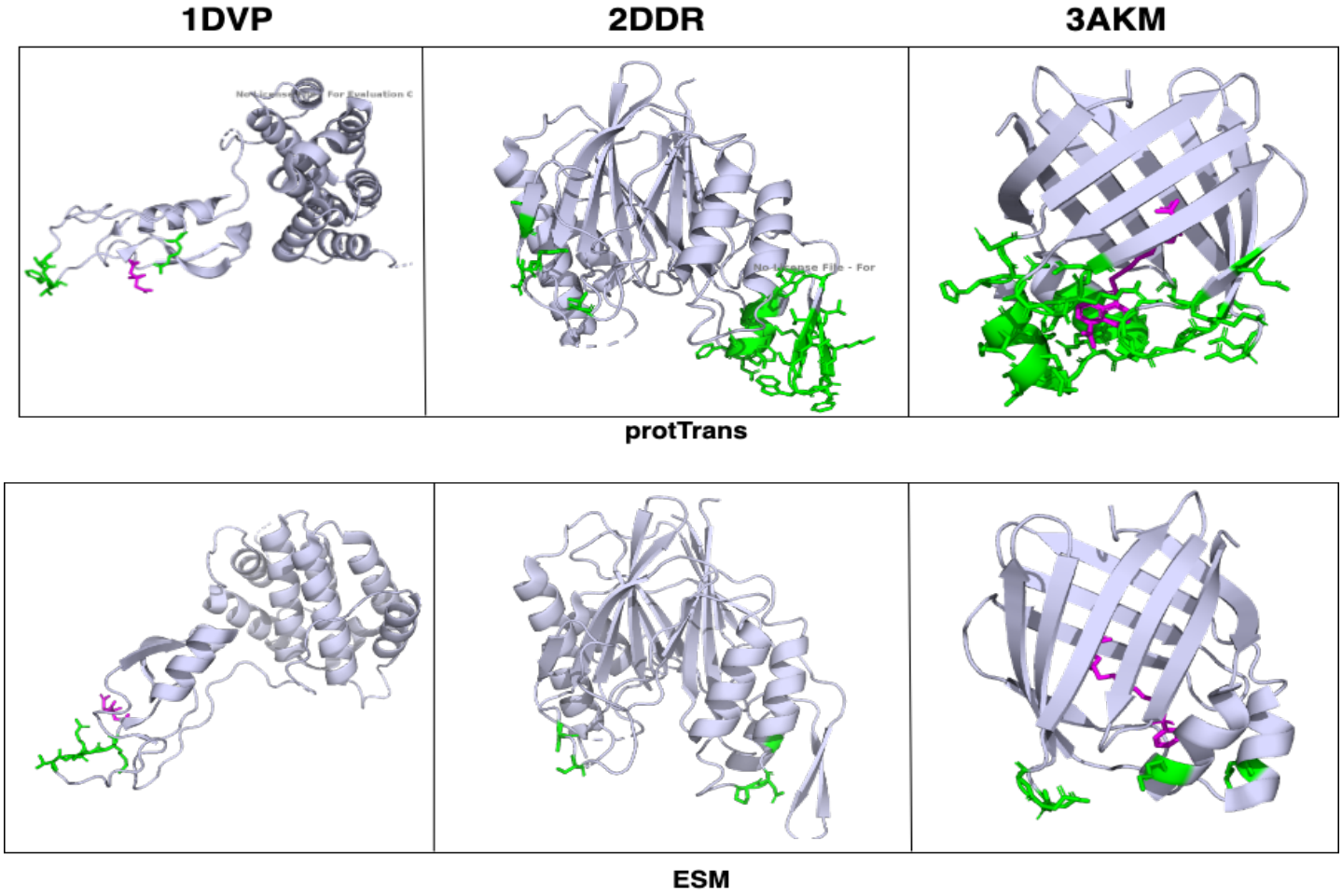


Figure 32: Predictions from the 2 MLP models for 3 proteins of the extra validation set, with experimental known protein-membrane interface regions.

Table 10: Peripheral membrane proteins of the extra validation set, their PDB ID, their experimentally known membrane-penetrating amino acids, and the predictions of our MLP models. Amino acid numbering is consistent with the PDB structure

Protein – PDB ID	Membrane penetrating regions / amino acids	protTrans MLP	ESM MLP
1c1z, 1coy, 1es6, 1s6x, 1tqn, 2ayl, 2mh1, 4x08, 5f0p	Not prediction from our models	-	-

1dvp	F173	T172, F173, T174, R181	T174, N175, R176, K177
1ffj	L6, V7, P8, L9, F10, Y22, M24, F25, M26, V27, P30, V32, P33, V34, I39, L47, L48, V49	L6, V7, P8, L9, F10, S11, T13, M26, V27, A28, A29, P30, H31, V32, K35, L47, L48	K5, A29, P30, V32
1gyg	Y331, F334	D216, G266, E267, K268, D269, A270, G271, D273, G296, N297, D298, M300, T301, K330, Y331, T332, A333, F334, P335, D336, A337	S265, G266, E267, K268, D269, A270, G271, T272, D273, N297, K330, Y331, T332, A333, F334, P335, D336
1h0a	L6, M10, I13, V14	-	R114
1iaz	W112, Y113	-	P81, Y110, N139
1joc	V1367, T1368, V1369	S1366, V1367, T1368, V1369	T1368, V1369
1nl1	F5, L6, V9	K97, E99, T103, T104	-
1oiz	F165, F169, I202, V206, M209	F165, A168, F169, P200, V201, I202, F203, H204, A205, V206, S208, M209, I210, F213	-
1pfo	W466, T490, L491	T460, L462, A463, E465	S190
1vfy	L185, L186	K182, S184, L185, L186	S173, K181, K182, L186, N187, R188, K189
2ddr	W284, F285	L90, N92, Y93, T97, N237, I239, A240, K241, Y242, N243, F244, P245, D246, W279, V281, T282, S283, W284, F285, Q286, K287, Y288, T289, D292	P26, N27, G29, S94, S96
2fnq	W413, F414, Y448, W449	D384, R385, E386, H387, A388, G389, T390, D391, W413, F414, H415, N416, D417, E419, A420, G445, G446, G447, Y448, W449, D452, P453, D454	G383, D384, R385, E386, H387, A388, G389, T390, D391, H415, N416, D444, G445, G446, G447, Y448, W449, D452, P453, K854

2p0m	Y15, F70, L71, W181, L195	S13, I14, Y15, A16, G17, K19, H69, F70, L71, K72, E73, D74, A75	G11, A12, S13, I14, A16, G17, S18, K19, K68, H69, F70, L71, K72, E73, D74
3akm	Around amino acid K27	K16, M18, E19, K20, M21, G22, V23, N24, I25, V26, K27, R28, K29, L30, A31, H33, D34, N35, A54, F55, R56, N57, N69, N71, A73, D74, G75, T76, E77, D97, N98, N100, E120	K20, K27, D97, N98, G99
3fsn	F196, F200, I202, F264, L265, W268, L270, W271	G48, V52, F57, I98, V99, I100, F108, F200, R234, F235, L261, W268, W271	-
3iiq	W300, W310	P87, L141	-
3jw8	Around amino acids L179, L186	E94, E96, S101, L184, R212, A213, V217	-
3npe	The two parallel amphipathic helices (α1:85- 109, α2:222-237)	P160, F162, D163, P164, V165, A166, G207, G216, G220, S222, R226, L367, L371, R372, E444, W501	-
3rzn	W142	V41, W142, I143, K146, I147, Y153	-
3w7r	Region 31-68	N129, P130, R131, P132	N145, F149, S151
5bzz	Regions 260- 269, 327-335	E40, R41, L42, E43, G44, V45, R74, K164, M205, F206, S207, G208, G209, T210, H259, Q261, N262, K263, M264, L265, K266, K267, D268, K330, R335	S207, G208, G209, T210, Q261, K269
5hxw	I345, L347, L351, I352, F355	-	N362, V366
6bfg	Region 177- 215	K17, K21, M22, V23, S178, A181, S214	S111, R167, D168, K174, I175, P176, S224

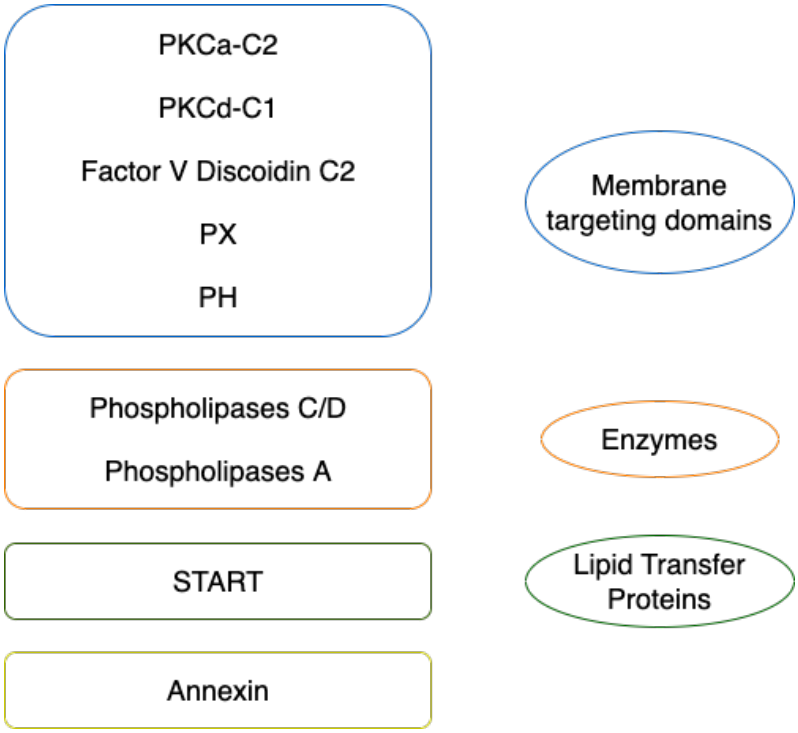


Figure 33: Protein domains belonging to nine protein superfamilies that represent the diversity of PMP functions: membrane targeting domains (PH, PKCa-C2, Factor V Discoidin C2, PKCd-C1, PX), enzymes (phospholipase C/D, phospholipase A), lipid transfer proteins (START), and annexins.

3.9 Predicting Protein-Membrane Interactions of Peripheral Membrane Proteins Using Attention Heads

Since the MLP classifiers have shown to be effective only in certain protein families, we aimed to enhance our results by developing a more generalized classifier. To this end, we proceeded to extract the attention maps of the two pre-trained language models. Initially, we sought to determine the specific hidden layers and attention heads where information pertaining to the interactions of amino acids with the membrane is encoded. Subsequently, we constructed a Logistic Regression model, which was trained on the attention maps, to classify whether an amino acid is membrane-penetrating or not.

3.9.1 Extraction

Each pLM consists of a different number of hidden layers and attention heads. More specifically, the protTrans model has 24 hidden layers and 32 attention heads in each layer – in total 768 attention maps, while the ESM model has 33 hidden layers and 20 attention heads in each layer – in total 660 attention maps. The attention maps vector size is analogous to the given protein at each time and the dimensions are *sequence_length* × *sequence_length*. Every row and every column represents each amino acid with a weight that indicates the importance or relevance of each amino acid residue in the protein sequence with respect to a particular residue [81], [101]. If we have for example the protein with Uniprot ID ‘A3DK57’ that has 831 amino acids, each attention map will be 831×831 (Figure 34). In addition, each attention map is a set of attention weights α for an input, where $\alpha_{i,j} > 0$ is the attention from token i (row) to token j

(column), such that $\sum_j a_{i,j} = 1$ [101]. The extraction of attention maps was done through the Hugging Face[5] library, by setting the parameter *output_attentions* equal to True in both models.

Sequence length x Sequence length

	M	R	K	...	T	N	
M	0.0004	0.00392	0.00392	...	0.0245	0.00028	Sum = 1
R	0.00999	0.087	0.004	...	0.00001	0.0567	Sum = 1
K	0.00312	0.00069	0.00297	...	0.034	0.0001	Sum = 1
...	...	...	...	...	...	...	
T	0.00007	0.00013	0.00993	...	0.0001	0.00044	Sum = 1
N	0.0075	0.00044	0.0097	...	0.00084	0.05	Sum = 1

Figure 34: An example of an attention map of A3DK57 protein with dimensions 831x831 and sum of rows equal to 1.

3.9.2 Interpretation of attention heads information

To manage decode and interpret the information that is passed among the attention maps, we used the following mathematical formula that is described in [101]:

$$\boldsymbol{p(f)} = \frac{\sum_{x \in X} \sum_{i=1}^{|x|} \sum_{j=1}^{|x|} \boldsymbol{f(i,j)} * \mathbb{1}_{\boldsymbol{a_{i,j} > \theta}}}{\sum_{x \in X} \sum_{i=1}^{|x|} \sum_{j=1}^{|x|} \mathbb{1}_{\boldsymbol{a_{i,j} > \theta}}} \quad (9)$$

where x is each protein, $\alpha_{i,j}$ is the attention weight in row i and column j, θ is a threshold to select for high-confidence attention weight and $f(i,j)$ which returns 1 if j is a membrane-penetrating amino acid and 0 otherwise. The formula (Eq. 9) computes the proportion of high-attention token pairs $(a_{i,j} > \theta)$ and it is possible to target the protein-membrane interface.

The total number of proteins that were used in our case was 30 and we calculated the proportion for 6 different thresholds (0.1, 0.15, 0.2, 0.3, 0.4, 0.5). Figures 35 & 36 depict the heatmaps for the 2 pLM that were generated using the formula (Eq. 9). As is shown on the heatmaps below, the information about the membrane penetrating amino acids for the protTrans model can be found in the last hidden layers and attention heads of the model in most of the thresholds, while in the ESM model, it is clearer, as the information is located on the first and the last hidden layers and in the middle attention heads in all the thresholds.

[5] https://huggingface.co/

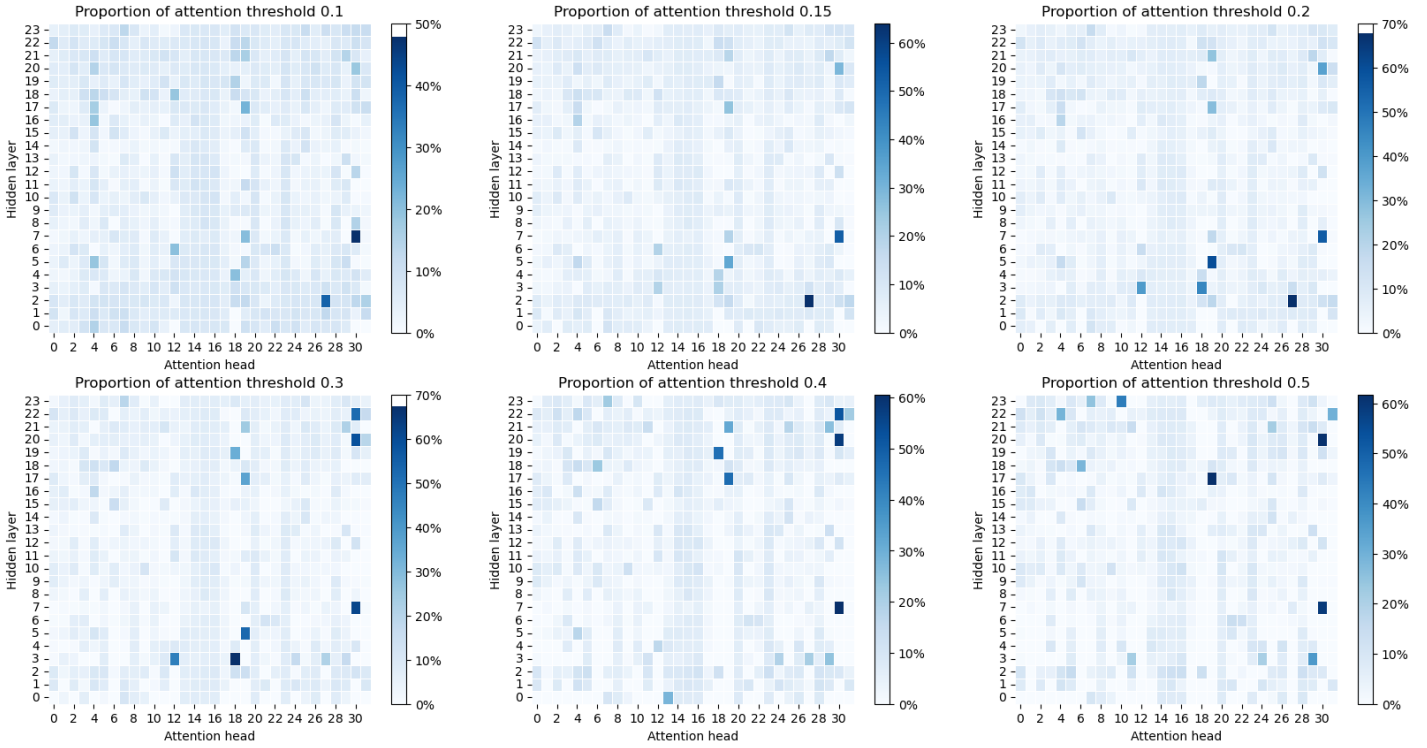


Figure 35: Heatmaps of attention maps that represent the proportion of attention in the several hidden layers with the respective attention heads for protTrans model based on different thresholds using a mathematical formula. The darker cells indicate the heads that map to the protein-membrane interface.

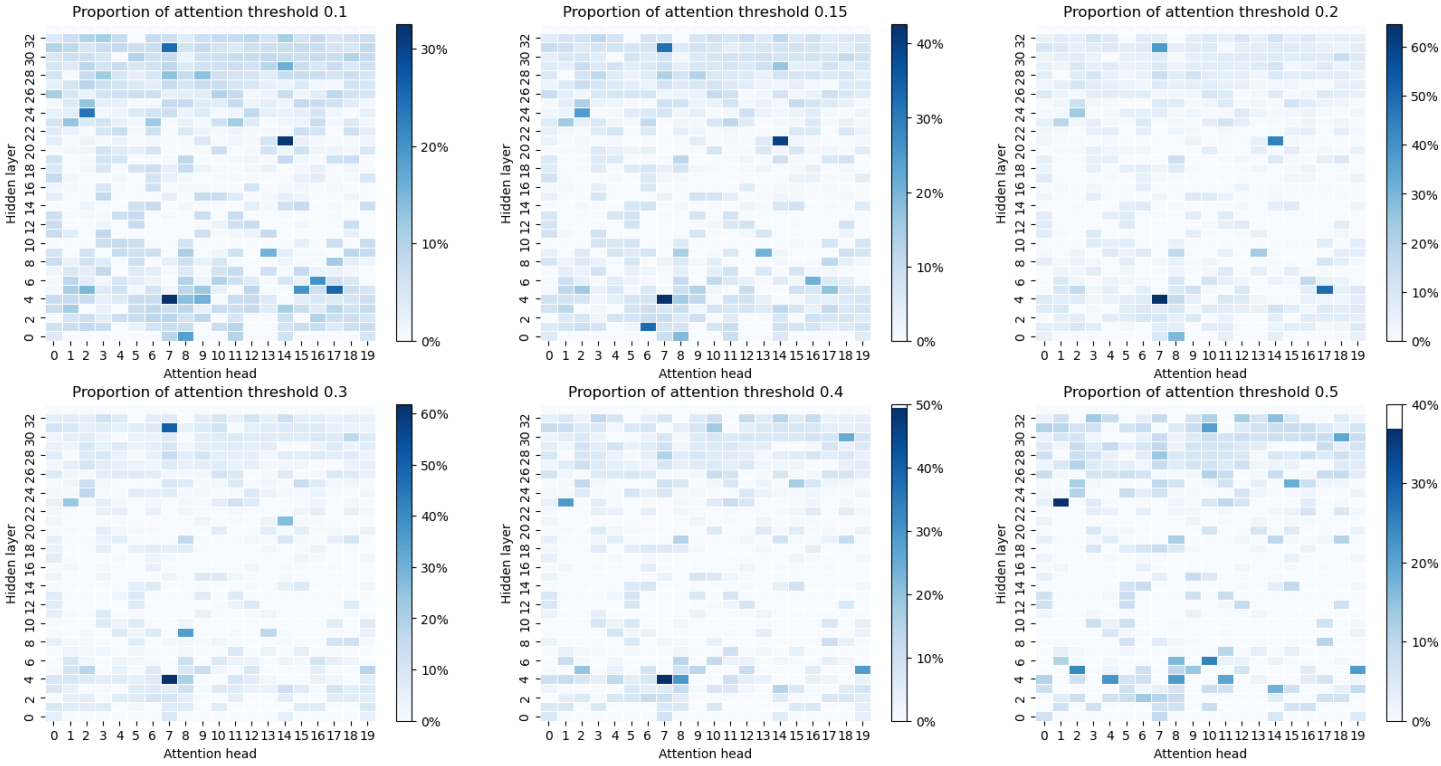


Figure 36: Heatmaps of attention maps that represent the proportion of attention in the several hidden layers with the respective attention heads for ESM model based on different thresholds using a mathematical formula. The darker cells indicate the heads that map to the protein-membrane interface.

3.9.3 Usage of attention heads to predict protein-membrane interactions

The approach used in this study was inspired by a previous work on ESM-1b, which extracted contacts from a transformer by passing a sequence through the model to obtain attention maps, and then applied logistic regression independently to each amino acid pair [87]. In this study, the attention maps of 20 proteins were extracted, and a dataset was created where the columns represent pairs of hidden layers with each attention head, the rows are the amino acid pairs of attention maps, and each cell indicates the weight of the amino acid pairs in the respective combination of hidden layer - attention head (Figure 37). The target column was created based on whether each amino acid was membrane-penetrating or not. The aim was to use this dataset to train a logistic regression model that could classify amino acids as membrane-penetrating or not based on the attention maps.

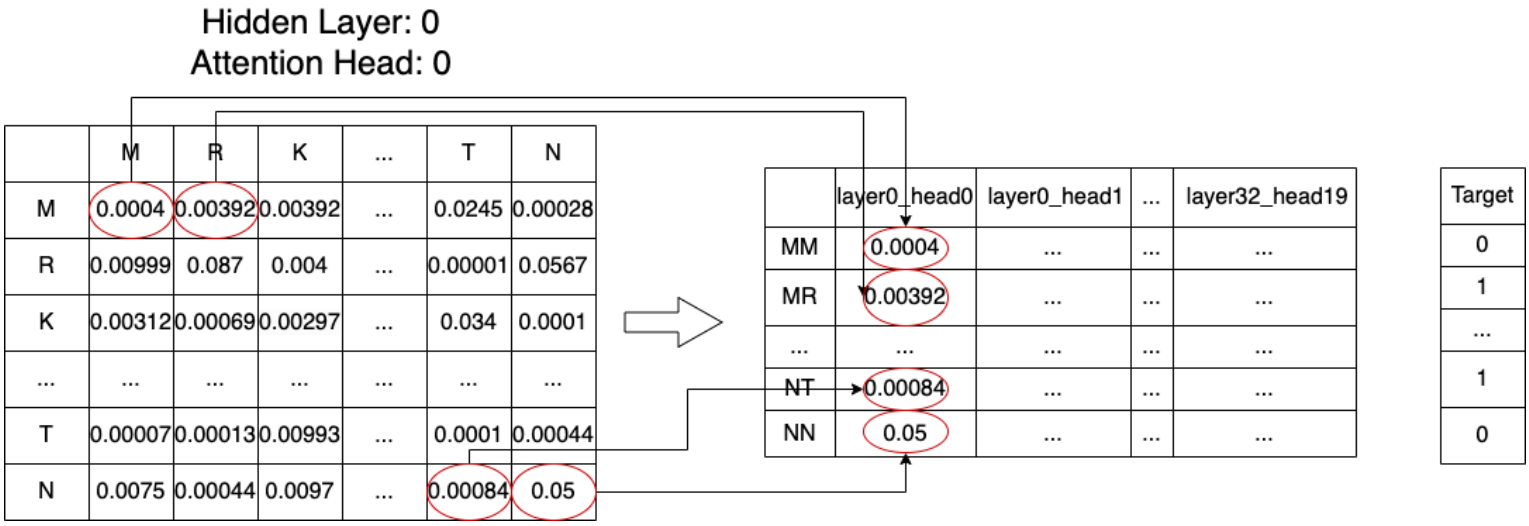


Figure 37: Example of how the dataset was generated for Logistic Regression based on the ESM model. First, the attention maps were extracted and then transferred to the aggregated table alongside the information about the labels of amino acids.

Having created the dataset, we then define the probability of a penetrating amino acid to a logistic regression with parameter β [87]:

$$\boldsymbol{p}\left(\boldsymbol{c}_{\boldsymbol{ij}}^{\boldsymbol{d}};\boldsymbol{\beta}\right)=\frac{\mathbf{1}}{\mathbf{1}+\boldsymbol{exp}\left(-\boldsymbol{\beta}_{\mathbf{0}}-\sum_{\boldsymbol{m}=\mathbf{1}}^{\boldsymbol{M}}\sum_{\boldsymbol{h}=\mathbf{1}}^{\boldsymbol{H}}\boldsymbol{\beta}_{\boldsymbol{mh}}\boldsymbol{a}_{\boldsymbol{mhij}}^{\boldsymbol{d}}\right)} \qquad (10)$$

where d is each protein, M is the layers, H is the heads, and a_{mhij} is the attention weight between the sequence position i and j in the given m-layer and h-head. The β will be the fitted parameter. In total, the model learns $MH + 1$ parameters, which many of them are zero because of the L_1 regularization.

Using the Stochastic Gradient Descend Classifier (SGDClassifier) from scikit-learn [63], we trained two models on the two different datasets (ESM & protTrans) applying L_1 regularization and *log_loss* as the loss function. For detecting which hidden layers and attention heads were used for the classification of the amino acids' pairs, we construct the heatmaps of models' coefficients (the absolute values), many of which are zero (Figure 38).

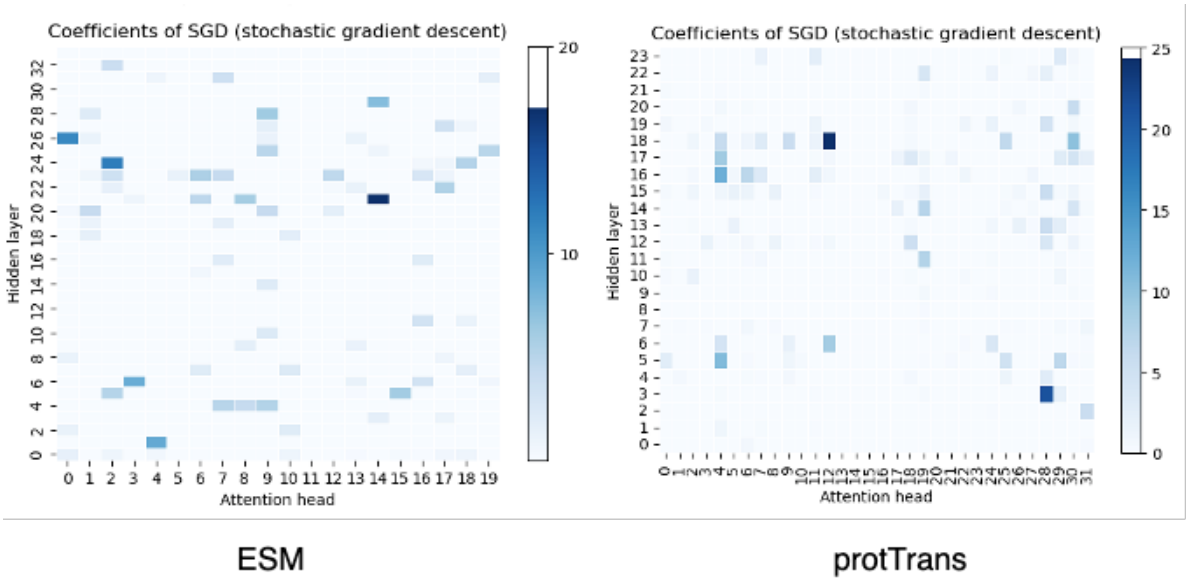


Figure 38: Heatmaps of SGDClassifiers coefficients which indicates the hidden layers and attention maps that were used for the classification.

To determine the models' performance, we initially needed to convert amino acid pair predictions into individual amino acid predictions by retaining the most frequently predicted class for each amino acid. The evaluation was carried out on five proteins, and the resulting outcomes are presented in Table 11 and Figure 39. We employed a range of thresholds while utilizing amino acid classification probabilities, and the outcomes presented herein used a 0.5 cut-off threshold. Despite our use of attention maps, we were unable to attain better outcomes, with numerous instances of incorrect classification, including roughly half of the amino acids being erroneously identified as membrane-penetrating and others predicting only one or two amino acids incorrectly. The same trend was observed with unknown protein families, resulting in similar results.

Table 11: Results of SGDClassifier on 5 test proteins in the 2 datasets.

	ESM	**protTrans**
F_1 score	0.35	0.366
MCC	0.341	0.344
Confusion matrix	**no_IBS IBS** **no_IBS** 2559 254 **IBS** 54 83	**no_IBS IBS** **no_IBS** 2629 184 **IBS** 65 72

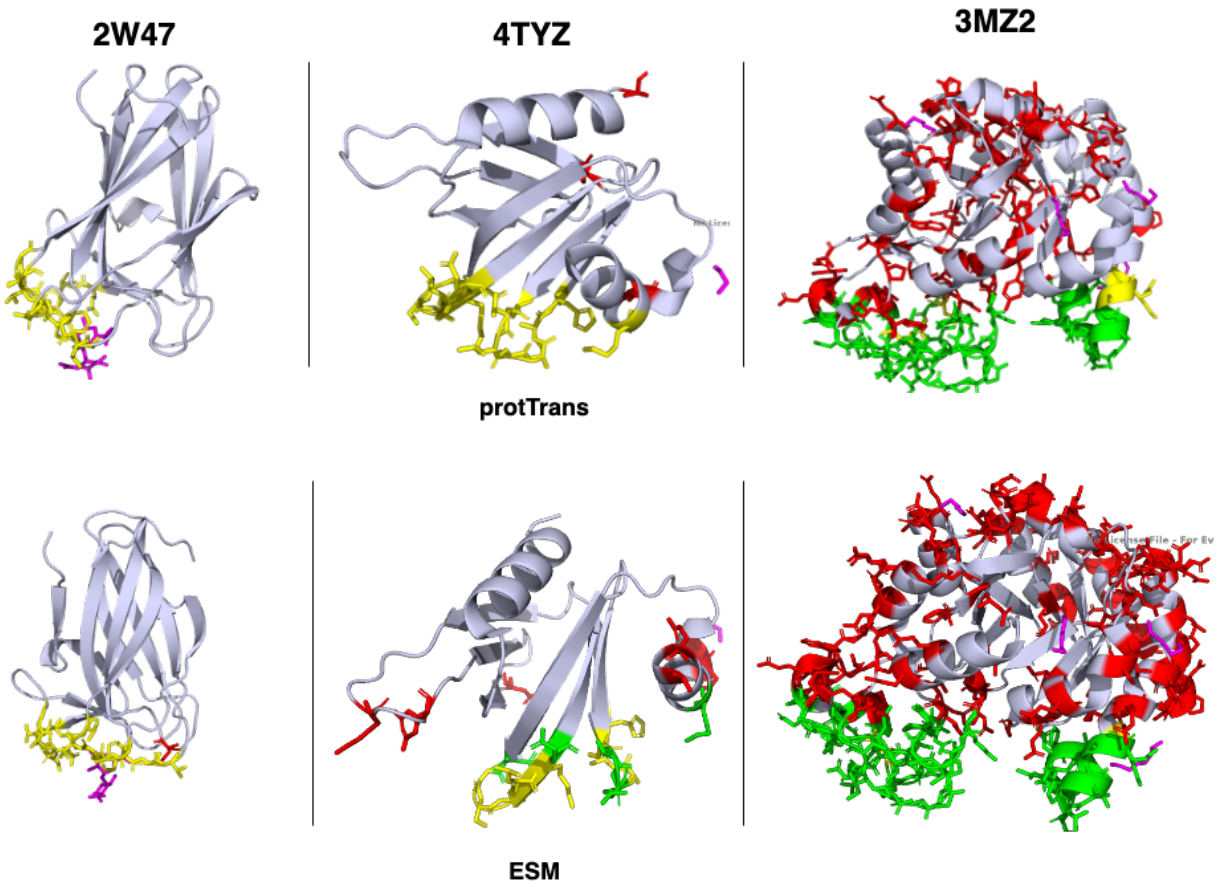


Figure 39: Predictions from the 2 SGDClassifiers models for 3 proteins of the test set.

4. DISCUSSION AND CONCLUSIONS

Characterizing interactions at the protein-membrane interface is crucial, as abnormal peripheral protein-membrane attachment is involved in the onset of many diseases. A limiting factor in studying and understanding protein-membrane interactions is that the membrane-binding domains of peripheral membrane proteins are typically unknown. Only a few methods exist that can predict protein-membrane interaction regions of peripheral membrane proteins from their 3D protein structure [40]–[43], and in many cases these methods are time-consuming. Moreover, the complexity of the interface and the lack of suitable procedures and modeling technologies have not allowed the routine employment of the protein-membrane interface in drug design pipelines.

By applying Artificial Intelligence (AI) techniques in the context of Natural Language Processing (NLP) the accuracy and prediction time for protein-membrane interface analysis can be significantly improved compared to existing methods. In this thesis, we describe a machine learning methodology for predicting membrane-penetrating amino acids using NLP and protein language models (pLMs). First, we assemble a dataset of 709 peripheral membrane proteins with experimentally established membrane-penetrating amino acids. We separate this dataset into two classes, the amino acids class that are membrane-penetrating and the amino acids class that is non-membrane-penetrating. We then extract the embeddings of amino acids from two protein Language Models (pLMs), ESM [96] and protTrans [84]; these are then set as the features of the dataset. Then, we split the dataset into training, validation, and test sets for evaluating model performance. To address the imbalance issue as only ~3% of the amino acids belong to the membrane-penetrating class, we determine the class weights using specific weights to emphasize the minority class. We then proceed to train several machine learning models, including neural networks, using default parameters. These models were employed to optimize the hyper-parameters of the MLP classifiers, which produced the highest scores during the initial stage of analysis for both datasets. For the hyper-parameter optimization, the Bayesian search technique was applied in an extensive range of values for 50 trials. The performance of the two models (Multi-Layer Perceptron trained on ESM features and Multi-Layer Perceptron trained on protTrans features) was improved compared to the initial test, yielding an F_1 score = 0.691 with MCC = 0.652 and F_1 score = 0.622 with an MCC = 0.577 for the protTrans and ESM features respectively, for correctly predicting the membrane-penetrating amino acids. Close inspection of the results revealed that many of the false positive predictions are true positives. Using a cutoff radius of 3Å around true positive membrane penetrating amino acids to include neighboring amino acids elevates the F_1 score by an average of 11%.

Although the MLP models demonstrated excellent speed and accuracy in predicting membrane-penetrating amino acids for proteins represented in the training set, their predictive capabilities are limited to this set alone. Specifically, the models cannot make predictions for proteins belonging to the transmembrane category or for those within protein families not included in the training set. As such, it is noteworthy that the performance of the MLP classifier models is highly reliant on the features/information utilized in this study. PLMs generate embeddings that contain encoded function and structural information of protein sequences, which is crucial in the decision-making

process of our models. This provides novel physicochemical insights into how peripheral membrane proteins interact with and adhere to the membrane. While the embeddings contain valuable information about the structural and functional properties of protein sequences, this wealth of information can sometimes be overwhelming and may result in relatively underwhelming results for the models. In the current task, features relative to the physicochemical properties of the proteins are more suitable, such as the hydrophobicity, solvent exposure surface area, secondary structure element, etc. [44]. So, extra work has to be carried out to explore whether it is possible to configure the embeddings with respect to the physicochemical properties of amino acids. This could result in a more targeted set of features that may boost the performance of models.

In addition, while MLP models are limited to specific protein families, they don't predict false amino acids if they cannot find any relevant information. This suggests that they are not overfitted, but they are also not very robust when it comes to unknown protein families. The fact that MLP models are limited to specific types of inputs can decrease the performance of the classifiers and requires further investigation.

Next, the attention maps were used to verify that the information about the protein-membrane interacting region is encoded within the pLMs [101] and to develop classifiers that are better able to classify membrane-penetrating amino acids [87]. The results confirmed the existence of interfacial binding site information in the attention heads of pLM's hidden layers. However, it was observed that either the information of attention maps or the Logistic Regression algorithm is unsuitable for accurately classifying protein-membrane interaction tasks as they achieve an F_1 score = 0.366 with an MCC = 0.344 and F_1 score = 0.35 with an MCC = 0.341 for the protTrans and ESM attention maps respectively. We tested several probability thresholds for the classification of the amino acids, but the performance was not improved. Inspection of the predicted values showed that they tend to classify randomly the membrane-penetrating amino acids. It should also be noted that attention maps were used to predict the contacts of amino acids in [87], as they discovered a correlation between self-attention maps and contact patterns [101]. Considering the above, further research needs to be done before rejecting the use of attention maps in this case, as they appear to be promising [81].

Overall, our results indicate that Artificial Intelligence, specifically NLP techniques, can make a significant contribution in the research community to tasks related to protein-membrane interaction, resulting in a substantial reduction in computing time. Significantly, our MLP models trained on pLM features can predict protein-membrane amino acids within a minute, which is faster than other tools that may require over an hour to perform the same task. As language models continue to improve and outperform traditional algorithms, it is likely that their usage will increase in the future [108]–[110].

5. FUTURE PERSPECTIVES

A number of potential improvements can be suggested to increase efficiency and predictability of the current project.

To evaluate the performance of our model, we used metrics appropriate for the imbalance dataset we have available [60], [73], [74], [106]. However, while the F_1 score and the Matthews correlation coefficient can assess the outcomes of the models, they cannot capture the specificities of protein structure, such as the position of amino acids in the 3D space. For instance, the score metrics are incapable of detecting scenarios in which amino acids situated in the protein-membrane interface, adjacent to true positives or adjacent loops, and are categorized as membrane-penetrating despite being labeled as non-membrane-penetrating. Therefore, it would be more appropriate to construct metrics that take into account the protein 3D structure, omitting the need for manual inspection of each protein, and resulting in a more precise scrutinization of the outcomes.

Currently, our Multi-Layer Perceptron models perform well in predicting protein-membrane interfaces for specific families of peripheral membrane proteins, but they are not capable of handling unknown protein families Although our models do not provide wrong information about these proteins, they lack generalizability and robustness. To address this issue, we need more protein information from diverse protein families, which could significantly improve the performance of our models. We also recognize the importance of secondary and supersecondary structure information in the binding process of peripheral membrane proteins; for example, alpha-helices have a higher propensity to interact with the membrane. Thus, incorporating such specific information into the dataset could enhance the predictive ability of our models. In addition, in the protein clustering procedure, we selected a threshold of 40% for clustering the proteins. However, implementing a less stringent pairwise similarity cutoff could potentially enable the algorithms to acquire a more diverse range of protein sequences.

As the choice of dataset can have a substantial impact on model performance, careful selection of input data is essential. One potential solution is to use an alternative protein dataset sourced from a database such as BioGRID [111], which contains known protein interactions, to train machine learning algorithms. The use of this dataset is expected to be less biased and more informative than our current dataset, as it encompasses a broader range of protein families.

As testing and optimization of machine learning algorithms is a time-consuming procedure, in this project four machine learning classifiers were chosen for predicting the membrane-penetrating amino acids. The present study involved performing model fine-tuning on two distinct datasets, namely the ESM dataset and the protTrans dataset. The multilayer perceptron (MLP) model was subjected to fine-tuning on both datasets. Additionally, Bayesian optimization was utilized to fine-tune an XGBoost model specifically on the protTrans dataset, although no significant improvement was observed in comparison to the untuned model. Thus, a more detailed investigation can be performed on the algorithms that may be potentially used, as other methods could demonstrate better performance, such as Support Vector Machines (SVMs), K-Nearest Neighbor (KNN), Naive Bayes, or others.

ABBREVIATIONS – ACRONYMS

PMP	Peripheral Membrane Proteins
AI	Artificial Intelligence
TP	True Positive
FP	False Positive
FN	False Negative
TN	True Negative
MCC	Matthews correlation coefficient
NLP	Natural Language Processing
LM	Language Model
BERT	Bidirectional Encoder Representations from Transformers
PLM	Protein Language Model
ESM	Evolutionary Scale Modeling
NN	Neural Network
CNN	Convolution Neural Network
FNN	Feedforward Neural Network
PePrMInt	Resources for Peripheral Protein-Membrane Interactions
IBS	Interfacial Binding Sites
DREAMM	Drugging pRotein mEmbrAne Machine learning Method
PDB	Protein Data Bank
Trp	Tryptophan
Phe	Phenylalanine
Tyr	Tyrosine
Met	Methionine
Lys	Lysine
Arg	Arginine
Gly	Glysine
MLP	Multi-Layer Perceptron
MODA	Membrane Optimal Docking Area
PPM	Positioning of Proteins in Membrane

REFERENCES

[1] A. Kessel and N. Ben-Tal, *Introduction to Proteins*. Chapman and Hall/CRC, 2018.

[2] J. P. Overington, B. Al-Lazikani, and A. L. Hopkins, “How many drug targets are there?,” *Nat Rev Drug Discov*, vol. 5, no. 12, pp. 993–996, 2006.

[3] Z. Cournia *et al.*, “Membrane Protein Structure, Function, and Dynamics: a Perspective from Experiments and Theory,” *J Membrane Biol*, vol. 248, no. 4, pp. 611–640, 2015.

[4] D. M. Boes, A. Godoy-Hernandez, and D. G. G. McMillan, “Peripheral Membrane Proteins: Promising Therapeutic Targets across Domains of Life,” *Membranes (Basel)*, vol. 11, no. 5, 2021.

[5] “Representation of integral, peripheral and lipid-anchored proteins on a model cell membrane [image] Available at: https://www.creative-biolabs.com/blog/index.php/membrane-protein-overview/ [Accessed 17 Feb. 2023].”

[6] V. Monje-Galvan and J. B. Klauda, “Peripheral membrane proteins: Tying the knot between experiment and computation,” *Biochim Biophys Acta - Biomembr*, vol. 1858, no. 7, Part B, pp. 1584–1593, 2016.

[7] T. Kinoshita, “Glycosylphosphatidylinositol (GPI) anchors: Biochemistry and cell biology: Introduction to a thematic review series,” *J Lipid Res*, vol. 57, no. 1. 2016.

[8] G. U. Ebiloma, E. O. Balogun, E. J. Cueto-Díaz, H. P. de Koning, and C. Dardonville, “Alternative oxidase inhibitors: Mitochondrion-targeting as a strategy for new drugs against pathogenic parasites and fungi,” *Med Res Rev*, vol. 39, no. 5. 2019.

[9] A. M. Whited and A. Johs, “The interactions of peripheral membrane proteins with biological membranes,” *Chem Phys Lipids*, vol. 192, pp. 51–59, 2015.

[10] T. Harayama and H. Riezman, “Understanding the diversity of membrane lipid composition,” *Nat Rev Mol Cell Biol*, vol. 19, no. 5, pp. 281–296, 2018.

[11] M. Dathe *et al.*, “Peptide Helicity and Membrane Surface Charge Modulate the Balance of Electrostatic and Hydrophobic Interactions with Lipid Bilayers and Biological Membranes,” *Biochemistry*, vol. 35, no. 38, pp. 12612–12622, Jan. 1996.

[12] R. Gamsjaeger *et al.*, “Membrane binding of β2-glycoprotein I can be described by a two-state reaction model: an atomic force microscopy and surface plasmon resonance study,” *Biochem J*, vol. 389, no. 3, pp. 665–673, Jul. 2005.

[13] A. H. Larsen, L. H. John, M. S. P. Sansom, and R. A. Corey, “Specific interactions of peripheral membrane proteins with lipids: what can molecular simulations show us?,” *Biosci Rep*, vol. 42, no. 4, p. BSR20211406, Apr. 2022.

[14] B. Rogaski and J. B. Klauda, “Membrane-Binding Mechanism of a Peripheral Membrane Protein through Microsecond Molecular Dynamics Simulations,” *J Mol Biol*, vol. 423, no. 5, pp. 847–861, 2012.

[15] I. A. Prior, P. D. Lewis, and C. Mattos, “A Comprehensive Survey of Ras Mutations in Cancer,” *Cancer Res*, vol. 72, no. 10, pp. 2457–2467, May 2012.

[16] A. D. Cox, S. W. Fesik, A. C. Kimmelman, J. Luo, and C. J. Der, “Drugging the undruggable RAS: Mission Possible?,” *Nat Rev Drug Discov*, vol. 13, no. 11, pp. 828–851, 2014.

[17] J. Canon *et al.*, “The clinical KRAS(G12C) inhibitor AMG 510 drives anti-tumour immunity,” *Nature*, vol. 575, no. 7781, 2019.

[18] B. Vanhaesebroeck, J. E. Burke, and R. R. Madsen, “Precision Targeting of Mutant PI3Kα in Cancer by Selective Degradation,” *Cancer Discov*, vol. 12, no. 1, pp. 20–22, Jan. 2022.

[19] J. S. Knight *et al.*, “Ectonucleotidase-Mediated Suppression of Lupus Autoimmunity and Vascular Dysfunction,” *Front Immuno*, vol. 9, 2018.

[20] P. Zukowska, B. Kutryb–Zajac, M. Toczek, R. T. Smolenski, and E. M. Slominska, “The role of ecto-5′-nucleotidase in endothelial dysfunction and vascular pathologies,” *Pharmacol Rep*, vol. 67, no. 4, pp. 675–681, 2015.

[21] M. Milella *et al.*, “PTEN: Multiple Functions in Human Malignant Tumors,” *Front Oncol*, vol. 5, 2015.

[22] J. T. Bendor, T. P. Logan, and R. H. Edwards, “The Function of α-Synuclein,” *Neuron*, vol. 79, no. 6, pp. 1044–1066, 2013.

[23] K. Segers *et al.*, “Design of protein–membrane interaction inhibitors by virtual ligand screening, proof of concept with the C2 domain of factor V,” *Proc Natl Acad Sci USA*, vol. 104, no. 31, pp. 12697–12702, Jul. 2007.

[24] P. Man *et al.*, “Defining the Interacting Regions between Apomyoglobin and Lipid Membrane by Hydrogen/Deuterium Exchange Coupled to Mass Spectrometry,” *J Mol Biol*, vol. 368, no. 2, pp. 464–472, 2007.

[25] J. H. Hurley, “Membrane binding domains,” *Biochim Biophys Acta Mol Cell Biol Lipids*, vol. 1761, no. 8, pp. 805–811, 2006.

[26] P. C. Spiegel, S. M. Kaiser, J. A. Simon, and B. L. Stoddard, “Disruption of Protein-Membrane Binding and Identification of Small-Molecule Inhibitors of Coagulation Factor VIII,” *Chem Biol*, vol. 11, no. 10, pp. 1413–1422, Oct. 2004.

[27] A. Nawrotek *et al.*, “PH-domain-binding inhibitors of nucleotide exchange factor BRAG2 disrupt Arf GTPase signaling,” *Nat Chem Biol*, vol. 15, no. 4, pp. 358–366, 2019.

[28] J. E. Johnson and R. B. Cornell, “Amphitropic proteins: regulation by reversible membrane interactions (Review),” *Mol Membr Biol*, vol. 16, no. 3, pp. 217–235, Jan. 1999.

[29] M. A. Lemmon, “Membrane recognition by phospholipid-binding domains,” *Nat Rev Mol Cell Biol*, vol. 9, no. 2. 2008.

[30] T. Tubiana, I. Sillitoe, C. Orengo, and N. Reuter, “Dissecting peripheral protein-membrane interfaces,” *bioRxiv*, 2022.

[31] J. L. MacCallum, W. F. Drew Bennett, and D. Peter Tieleman, “Distribution of amino acids in a lipid bilayer from computer simulations,” *Biophys J*, vol. 94, no. 9, 2008.

[32] A. C. V. Johansson and E. Lindahl, “Position-resolved free energy of solvation for amino acids in lipid membranes from molecular dynamics simulations,” *Proteins: Struct Funct Genet*, vol. 70, no. 4, 2008.

[33] S. Jones and J. M. Thornton, “Principles of protein-protein interactions,” *PNAS*, vol. 93, no. 1. 1996.

[34] S. H. White, “Translocons, thermodynamics, and the folding of membrane proteins,” in *FEBS Letters*, 2003.

[35] F. Cymer, G. Von Heijne, and S. H. White, “Mechanisms of integral membrane protein insertion and folding,” *J Mol Biol*, vol. 427, no. 5. 2015.

[36] K. C. Nastou, G. N. Tsaousis, N. C. Papandreou, and S. J. Hamodrakas, “MBPpred: Proteome-wide detection of membrane lipid-binding proteins using profile Hidden Markov Models,” *Biochim Biophys Acta - Proteins Proteom*, vol. 1864, no. 7, 2016.

[37] Y. Sharikov *et al.*, “MAPAS: A tool for predicting membrane-contacting protein surfaces [4],” *Nat Methods*, vol. 5, no. 2. 2008.

[38] N. Bhardwaj, R. V. Stahelin, R. E. Langlois, W. Cho, and H. Lu, “Structural Bioinformatics Prediction of Membrane-binding Proteins,” *J Mol Biol*, vol. 359, no. 2, 2006.

[39] D. L. Scott, G. Diez, and W. H. Goldmann, “Protein-lipid interactions: Correlation of a predictive algorithm for lipid-binding sites with three-dimensional structural data,” *Theor Biol Medical Model*, vol. 3. 2006.

[40] M. A. Lomize, I. D. Pogozheva, H. Joo, H. I. Mosberg, and A. L. Lomize, “OPM database and PPM web server: Resources for positioning of proteins in membranes,” *Nucleic Acids Res*, vol. 40, no. D1, 2012.

[41] A. L. Lomize, S. C. Todd, and I. D. Pogozheva, “Spatial arrangement of proteins in planar and curved membranes by PPM 3.0,” *Protein Sci*, vol. 31, no. 1, 2022.

[42] I. Kufareva *et al.*, “Discovery of novel membrane binding structures and functions,” *Biochem Cell Biol*, vol. 92, no. 6, 2014.

[43] A. Chatzigoulas and Z. Cournia, “DREAMM: a web-based server for drugging protein-membrane interfaces as a novel workflow for targeted drug design,” *Bioinform*, vol. 38, no. 24, pp. 5449–5451, Dec. 2022.

[44] A. Chatzigoulas and Z. Cournia, “Predicting protein-membrane interfaces of peripheral membrane proteins using ensemble machine learning,” *Brief Bioinform*, vol. 23, no. 2, 2022.

[45] A. M. Turing, “Computing machinery and intelligence,” in *Parsing the Turing Test: Philosophical and Methodological Issues in the Quest for the Thinking Computer*, 2009.

[46] J. Liu *et al.*, “Artificial intelligence in the 21st century,” *IEEE Access*, vol. 6, 2018.

[47] Y. Zeng and L. Wang, “Fei-Fei Li: Artificial Intelligence is on its way to reshape the world,” *Natl Sci*, vol. 4, no. 3, 2017.

[48] S. Sikdar, “Artificial intelligence, its impact on innovation, and the Google effect,” *Clean Technol Environ Policy* , vol. 20, no. 1. 2018.

[49] S. Das, A. Dey, A. Pal, and N. Roy, “Applications of Artificial Intelligence in Machine Learning: Review and Prospect,” *Int J Comput Appl*, vol. 115, no. 9, 2015.

[50] M. I. Jordan and T. M. Mitchell, “Machine learning: Trends, perspectives, and prospects,” *Science*, vol. 349, no. 6245. 2015.

[51] J. E. van Engelen and H. H. Hoos, “A survey on semi-supervised learning,” *Mach Learn*, vol. 109, no. 2, 2020.

[52] “The 3 machine learning methods [image] Available at: https://www.altexsoft.com/blog/semi-supervised-learning/ [Accessed 19 Feb. 2023].”

[53] M. Batta, “Machine Learning Algorithms - A Review,” *IJSR*, vol. 18, no. 8, 2018.

[54] G. S. Chilla, L. Y. Yeow, Q. H. Chew, K. Sim, and K. N. B. Prakash, “Machine learning classification of schizophrenia patients and healthy controls using diverse neuroanatomical markers and Ensemble methods,” *Sci Rep*, vol. 12, no. 1, 2022.

[55] A. Halevy, P. Norvig, and F. Pereira, “The unreasonable effectiveness of data,” *IEEE Intell Syst*, vol. 24, no. 2, 2009.

[56] A. Paullada, I. D. Raji, E. M. Bender, E. Denton, and A. Hanna, “Data and its (dis)contents: A survey of dataset development and use in machine learning research,” *Patterns*, vol. 2, no. 11. 2021.

[57] A. Vabalas, E. Gowen, E. Poliakoff, and A. J. Casson, “Machine learning algorithm validation with a limited sample size,” *PLoS One*, vol. 14, no. 11, 2019.

[58] J. Cai, J. Luo, S. Wang, and S. Yang, “Feature selection in machine learning: A new perspective,” *Neurocomputing*, vol. 300, 2018.

[59] G. Chandrashekar and F. Sahin, “A survey on feature selection methods,” *Comput Electr Eng*, vol. 40, no. 1, pp. 16–28, 2014.

[60] A. Ali, S. M. Shamsuddin, and A. L. Ralescu, “Classification with class imbalance problem: A review,” *IJASCA*, vol. 7, no. 3, 2015.

[61] N. v. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” *J Artif Intell Res*, vol. 16, 2002.

[62] H. He, Y. Bai, E. A. Garcia, and S. Li, “ADASYN: Adaptive synthetic sampling approach for imbalanced learning,” in *IEEE IJCNN*, 2008.

[63] F. Pedregosa *et al.*, “Scikit-learn: Machine learning in Python,” *J Mach Learn Res*, vol. 12, 2011.

[64] F. Mercaldo, V. Nardone, and A. Santone, “Diabetes Mellitus Affected Patients Classification and Diagnosis through Machine Learning Techniques,” *Procedia Comput Sci*, vol. 112, 2017.

[65] T. Kaddoura *et al.*, “Acoustic diagnosis of pulmonary hypertension: Automated speech- recognition-inspired classification algorithm outperforms physicians,” *Sci Rep*, vol. 6, 2016.

[66] X. Jiang, Y. Wang, W. Liu, S. Li, and J. Liu, “CapsNet, CNN, FCN: Comparative performance evaluation for image classification,” *IJML*, vol. 9, no. 6, 2019.

[67] S. R. Safavian and D. Landgrebe, “A Survey of Decision Tree Classifier Methodology,” *IEEE Trans Syst Man Cybern*, vol. 21, no. 3, 1991.

[68] A. K. Jain, R. P. W. Duin, and J. Mao, “Statistical pattern recognition: A review,” *TPAMI / PAMI*, vol. 22, no. 1, 2000.

[69] C. Gambella, B. Ghaddar, and J. Naoum-Sawaya, “Optimization Models for Machine Learning: A Survey,” *arXiv preprint arXiv:1901.05331*, 2019.

[70] J. Bergstra and Y. Bengio, “Random search for hyper-parameter optimization,” *J Mach Learn Res*, vol. 13, 2012.

[71] J. Snoek, H. Larochelle, and R. P. Adams, “Practical Bayesian optimization of machine learning algorithms,” in *Adv Neur In*, 2012.

[72] D. Passos and P. Mishra, “A tutorial on automatic hyperparameter tuning of deep spectral modelling for regression and classification tasks,” *Chemometr Intell Lab*, vol. 223. 2022.

[73] S. Boughorbel, F. Jarray, and M. El-Anbari, “Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric,” *PLoS One*, vol. 12, no. 6, 2017.

[74] Y. Tang, Y. Q. Zhang, and N. v. Chawla, “SVMs modeling for highly imbalanced classification,” *IEEE Trans Syst Man Cybern*, vol. 39, no. 1, 2009.

[75] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, “Learning word vectors for sentiment analysis,” in *ACL-HLT 2011*, 2011.

[76] S. Lai, L. Xu, K. Liu, and J. Zhao, “Recurrent convolutional neural networks for text classification,” in *Proc Innov Appl Artif Intell Conf*, 2015.

[77] B. McCann, J. Bradbury, C. Xiong, and R. Socher, “Learned in translation: Contextualized word vectors,” in *Adv Neur In*, 2017.

[78] E. H. Houssein, R. E. Mohamed, and A. A. Ali, “Machine Learning Techniques for Biomedical Natural Language Processing: A Comprehensive Review,” *IEEE Access*, vol. 9. 2021.

[79] J. Lee *et al.*, “BioBERT: A pre-trained biomedical language representation model for biomedical text mining,” *Bioinformatics*, vol. 36, no. 4, 2020.

[80] C. C. Huang and Z. Lu, “Community challenges in biomedical text mining over 10 years: Success, failure and the future,” *Brief Bioinform*, vol. 17, no. 1, 2016.

[81] A. Vaswani *et al.*, “Attention is all you need,” in *Adv Neur In*, 2017.

[82] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *NAACL HLT 2019*, 2019.

[83] C. Raffel *et al.*, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *J Mach Learn Res*, vol. 21, 2020.

[84] A. Elnaggar *et al.*, “ProtTrans: Towards Cracking the Language of Life’s Code Through Self-Supervised Learning,” *TPAMI / PAMI*, vol. 14, no. 8, 2021.

[85] A. Elnaggar *et al.*, “ProtTrans,” *bioRxiv*, vol. 14, no. 8, 2020.

[86] R. Rao *et al.*, “Evaluating protein transfer learning with TAPE,” in *Adv Neur In*, 2019.

[87] R. Rao, J. Meier, T. Sercu, S. Ovchinnikov, and A. Rives, “Transformer protein language models are unsupervised structure learners,” *bioRxiv*. 2020.

[88] R. Rao *et al.*, “MSA Transformer,” *bioRxiv*, 2021.

[89] B. E. Suzek, Y. Wang, H. Huang, P. B. McGarvey, and C. H. Wu, “UniRef clusters: A comprehensive and scalable alternative for improving sequence similarity searches,” *Bioinformatics*, vol. 31, no. 6, 2015.

[90] C. A. Orengo, A. D. Michie, S. Jones, D. T. Jones, M. B. Swindells, and J. M. Thornton, “CATH - A hierarchic classification of protein domain structures,” *Structure*, vol. 5, no. 8, 1997.

[91] M. Steinegger, M. Mirdita, and J. Söding, “Protein-level assembly increases protein sequence recovery from metagenomic samples manyfold,” *Nat Methods*, vol. 16, no. 7, 2019.

[92] K. Clark, M.-T. Luong, Q. v Le, and C. D. Manning, “ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators.” arXiv, 2020.

[93] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, “ALBERT: A Lite BERT for Self-supervised Learning of Language Representations.” arXiv, 2019.

[94] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. v. Le, and R. Salakhutdinov, “Transformer-XL: Attentive language models beyond a fixed-length context,” in *ACL 2019*, 2020.

[95] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. v. Le, “XLNet: Generalized autoregressive pretraining for language understanding,” in *Adv Neur In*, 2019.

[96] Z. Lin *et al.*, “Evolutionary-scale prediction of atomic level protein structure with a language model,” *bioRxiv*, 2022.

[97] J. Jumper *et al.*, “Highly accurate protein structure prediction with AlphaFold,” *Nature*, vol. 596, no. 7873, 2021.

[98] M. Baek *et al.*, “Accurate prediction of protein structures and interactions using a three-track neural network,” *Science (1979)*, vol. 373, no. 6557, 2021.

[99] “Attention heads paying attention to different parts [image] Available at: https://zhiyuchen.com/2020/01/05/paper-reading-what-does-bert-look-at-an-analysis-of-berts-attention/ [Accessed 20 Feb. 2023].”

[100] “Input of attention layer in the form of Query, Key & Value and representation of Multi-head attention [image] Available at: https://towardsdatascience.com/transformers-explained-visually-part-2-how-it-works-step-by-step-b49fa4a64f34 [Accessed 20 Feb. 2023].”

[101] J. Vig, A. Madani, L. R. Varshney, C. Xiong, R. Socher, and N. F. Rajani, “BERTology Meets Biology: Interpreting Attention in Protein Language Models.” arXiv, 2020.

[102] A. Bateman, “UniProt: A worldwide hub of protein knowledge,” *Nucleic Acids Res*, vol. 47, no. D1, 2019.

[103] W. Li and A. Godzik, “Cd-hit: A fast program for clustering and comparing large sets of protein or nucleotide sequences,” *Bioinform*, vol. 22, no. 13, 2006.

[104] H. M. Berman *et al.*, “The Protein Data Bank,” *Nucleic Acids Res*, vol. 28, no. 1. 2000.

[105] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in *KDD*, 2016.

[106] G. Lemaître, F. Nogueira, and C. K. Aridas, “Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning,” *J Mach Learn Res*, vol. 18, 2017.

[107] F. Chollet, “Keras: The Python Deep Learning library,” *Keras.Io*, 2015.

[108] K. S. Kalyan, A. Rajasekharan, and S. Sangeetha, “AMMU: A survey of transformer-based biomedical pretrained language models,” *J Biomed Inform*, vol. 126. 2022.

[109] N. Al-twairesh, “The evolution of language models applied to emotion analysis of arabic tweets,” *Information (Switzerland)*, vol. 12, no. 2, 2021.

[110] T. B. Brown *et al.*, “Language models are few-shot learners,” in *Adv Neural Inf Process Syst*, 2020.

[111] R. Oughtred *et al.*, “The BioGRID database: A comprehensive biomedical resource of curated protein, genetic, and chemical interactions,” *Protein Sci*, vol. 30, no. 1, 2021.