University of Padova

Department of Mathematics

Master Thesis in Data science

Machine learning models for the discovery of new plastics degradation enzymes

Supervisor
Guido Zampieri
University of Padova

Co-supervisor

Master Candidate
Stefano Minto

Academic Year
2023-2024

Dedication.
Khoya andiamo avanti

Abstract

Plastic pollution has emerged as a global environmental challenge, prompting the need for innovative strategies to address the mounting accumulation of plastic waste, such as bioremediation through living organisms like bacteria, fungi, or plants to break down or neutralize pollutants in the environment. This thesis explores the promising avenue of plastic degradation through microbial action, focusing on the search for microbial enzymes capable of breaking down plastics, with a particular emphasis on polyethylene terephthalate (PET). The goal of this work is to develop machine learning models able to identify enzymes for PET degradation in a pool of available proteins. Protein sequence and structure serve as complementary sources of information for creating numerical representations for each protein under analysis. These numerical representations are then used to train semi-supervised classification models capable of distinguishing PET-degrading proteins from others. Experimental validations on a representative protein set yield high performances for all the tested models, particularly those that incorporate sequence information. The results suggest that these methods can detect crucial molecular markers associated with the ability to degrade PET in both information sources, allowing the prediction of unknown PET-degrading enzymes coming from microorganisms adapted in heavily plastic-polluted environments.

Contents

Listing of figures

Listing of tables

Listing of acronyms

MSA . . . . . Multiple Sequence Alignment

Mcc . . . . . Matthew's correlation coefficient

AA . . . . . Amino acid

AAs . . . . . Amino acids

1 Introduction

In the last few years, the handling of plastic waste problem became more and more important since the usage of that material increases rapidly because of its low cost, versatility, and durability. The mass production of plastics began in the 1950s and annual production levels now exceed 380 million tons[5].
Plastic often has short service lifespans and, unfortunately, only a small fraction is recycled (roughly 9%), the remaining part is incinerated(12%) or is accumulated in landfills and natural environments (79%)[5]. The vast majority of monomers used to make plastics, such as propylene and ethylene, are derived from fossil hydrocarbons. The most commonly used plastics are not biodegradable, as a result, they accumulate rather than decompose in the natural environment[6]. It is today clear that plastic causes adverse effects in all ecosystems[7].

Environmental bio degradation of most conventional plastics, including polyethylene (PE), polypropylene (PP), polystyrene (PS) and polyethylene terephthalate (PET) has not been observed to any significant degree[7], therefore the scientific community is trying to discover new microorganisms and enzymes capable of biodegrading plastics, in fact for the past several decades there has been considerable interest in identifying plastic-degrading microorganisms and plastic-degrading enzymes. Exposure to plastic is a new occurrence for microorganisms, as these chemical compounds have only been introduced to bacteria in recent times, despite their evolution over millions of years. In particular, this thesis project is focused on the discovery of new proteins associated with PET degradation.

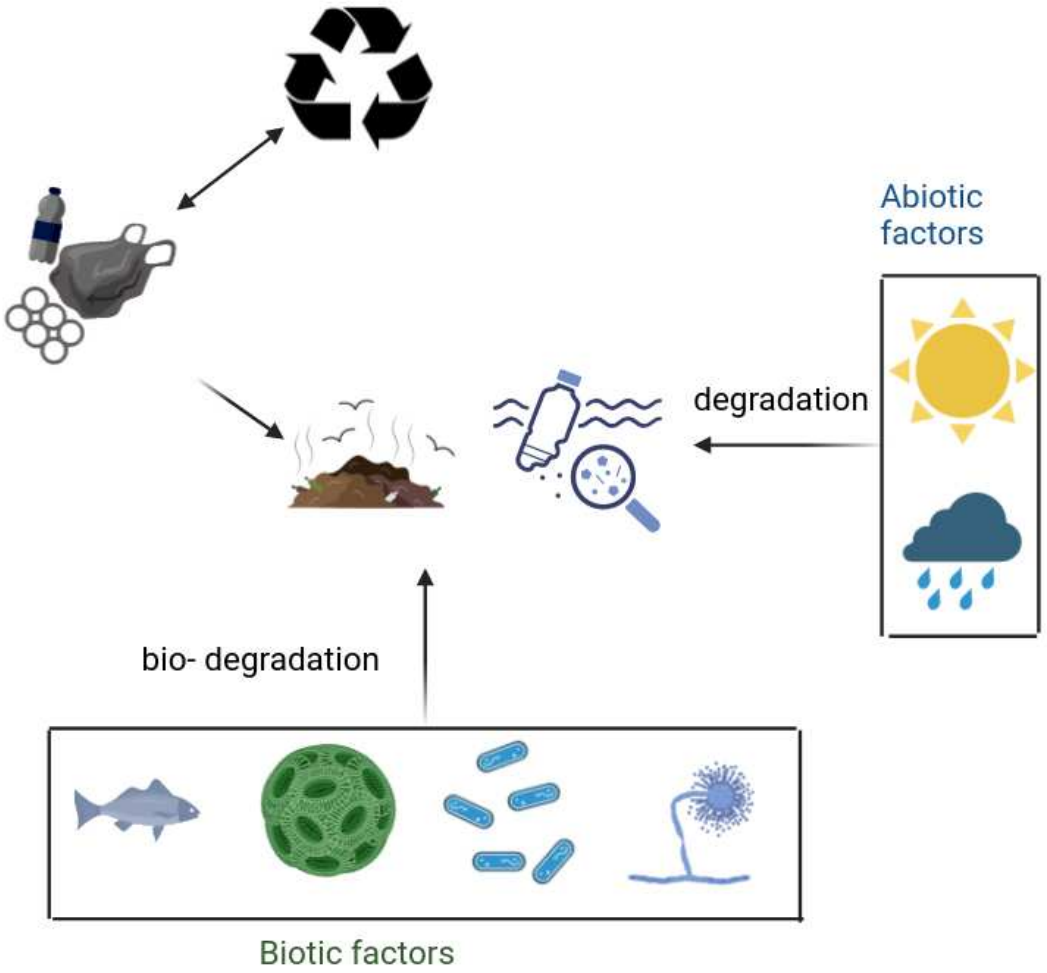

Figure 1.1: Although a portion of plastic is efficiently recycled an important of it reach the environment where it can be degraded by both biotioc and abiotic factors.

1.1 Plastic degradation

As we can see from the Figure 1.1, plastic degradation takes into account abiotic and biotic factors. Any physical or chemical change in polymer as a result of environmental factors such as light, heat, moisture, chemical conditions, or biological activity processes inducing changes in polymer properties. Most plastics tend to absorb high-energy radiation in the ultraviolet portion of the spectrum ($\sim$ 295-400nm), which activates their electrons to higher reactivity and causes oxidation, cleavage, and other degradation[8]. It is important to distinguish between deterioration of the bulk plastic (e.g. fragmentation resulting in microplastics) and depoly-

merization (degradation of the polymer at the molecular level). Normally, degradation trials involve the incubation of polymers in situ to investigate how plastics behave and/or degrade in different environments over time, which is largely dependent on polymer properties and environmental conditions[8].
A significant drawback of abiotic factors such as sunlight and rainfall is their slow rate of plastic degradation, regard to that it has been estimated that the degradation of a plastic bottle takes around 100 years[8]. Biodegradation is the process by which organic substances are broken down by living organisms. The term is often used in relation to ecology, waste management, and environmental remediation (bioremediation). Plastics are biodegraded aerobically in nature, anaerobically in sediments and landfills, and partially anaerobically in composts and soil[7]. Carbon dioxide (CO_2) and water are produced during aerobic biodegradation. Carbon dioxide, water and methane (CH_4) are instead produced during anaerobic biodegradation[9].
Generally, the breakdown of large polymers into carbon dioxide (mineralization) requires several different organisms, with one breaking the polymer into its constituent monomers, one using the monomers and excreting simpler waste compounds, and one able to use the excreted wastes. Standardized tests are used to certify polymers as biodegradable or compostable, in particular, plastic must reach a 90% conversion to CO_2 under specified conditons within 6 moths to be considered compostable[10].
Degradation studies can be black-box with no attempt to characterize the microbial community involved or can be paired with microbial community to identify plastic degraders, quantifying plastic degradation is important for the characterization of plastic degradation enzymes and can be done simultaneously with microbial community profiling in situ or under controlled laboratory conditions.
The identification of new plastic degradation agents has the specific goal of finding and characterizing new enzymes and/or microbial lineages that mediate plastic degradation.

1.2 Biodegradation of plastic

Some microrganisms such as bacteria, fungi, and algae, are involved in synthetic plastic degradation[9], the biodegradation of plastics proceed differently according to different soil conditions because the microorganisms responsible for the degradation differ from each other and they have their own optimal growth conditions in the soil.
Biodegradation is influenced by different factors that include polymer characteristics, type of organism, and nature of pretreatment[9]. During the degradation, the polymer is first con-

verted to its monomers, and then theese monomers are mineralized. Because most polymers are too large to pass through cellular membranes, they must first be depolymerized to smaller monomers before they can be absorbed and biodegraded within microbial cells[8].
Generally, an increase in molecular weight results in a decrease in the degradability of the polymer by microorganisms, in contrast, monomers, dimers and oligomers of the repeating units of a polymer are much easier to degrade and mineralize[8].

1.3 PET and its possible biodegradation

Polyethylene terephthalate (PET) is a polymer synthesized from repeating units of ethylene terephthalate. Each molecule of PET consists of two main monomer units :

1. **Ethylene Glycol (EG):** which is a diol compound (e.g. it contains two hydroxyl functional groups – OH) with chemical formula : $C_2H_6O_2$.

2. **Terephtalatic Acid (TPA):** which is an aromatic dicarboxylic acid with a chemical formula:
   $(C_6H_6(COOH)_2)$.

The most important features of this material are the following[11] :

- Property : PET offers several desirable properties for industrial applications, such as :

  1. Transparency : PET is transparent, allowing consumers to see the contents of the packaging.
  2. Strength : PET has excellent mechanical strength, making it suitable for packaging products that require protection during handling and transportation.
  3. Barrier Properties : PET provides good barrier properties against moisture, oxygen, and other gases, helping to extend the shelf life of packaged products. These characteristics result in minimal natural degradation of PET over time.
  4. Recyclability: PET is highly recyclable and can be recycled into new PET products or other materials, such as fibers for textiles and classical plastic bottles.

- Biodegradability : Enzymatic degradation : Enzymes such as PETase and MHETase, discovered[12] in bacteria Ideonella sakaiensis, have been found to catalyze the breakdown of PET into its constituent monomers, ethylene glycol (EG) and terephtalic acid (TPA), which can then be metabolized by microorganisms as a source of energy.

In recent decades, research activity on biodegradation has increased a lot. There are several biological organisms involved in this process, in particular bacteria, fungi, and microalgae. It has been proved[8] that the species belonging to the genus Bacillus are particularly good at degrading PET efficiently respect other microorganisms. In particular, Bacillus cereus and Bacillus gottheilii have been shown to adapt to other polymers, such as polyethylene (PE), polypropylene (PP), and polystyrene (PS). Together, the results[8] indicate that these microorganisms possess specific enzymatic mechanisms for the transformation of polymers into simpler forms that are ideal as an energy source for them. The ability of Bacillus sp. to utilize these substrates as a source of carbon and energy is evident in its adaptation to PET-contaminated environments.
During polymer degradation, microbes first adhere to the polymer surface, thereby exposing it to microbial colonization. Polymer colonization is followed by the secretion of extracellular enzymes, which bind to the polymer and cause hydrolytic cleavage, The polymer is subsequently degraded into low-weight polymers and mineralized to carbon dioxide (CO_2) and water (H_2O), which are used by the microbe as an energy source.

1.3.1 Identification of PET-degrading enzymes

In an era where data acquisition and storage is exponentially increasing, a revolution in data accessibility is underway in the biological field. In particular, instruments like nucleic acid sequencers are becoming much more efficient. A sequencer is an instrument that is used to determine the precise order of nucleotide in a DNA or RNA molecule or the order of amino acids in a protein. The continuous development of these instruments allows researchers to obtain sequences of an unknown protein rapidly and economically. For this reason, a large number of amino acid sequences are available but with undefined functions. Databases such as UniProt[13] contain manually reviewed protein sequences, where roles and functions are well defined, and unreviewed proteins whose functions remain undefined or predicted but not experimentally validated. Often, function prediction is based on sequence similarity to the reviewed proteins through the use of alignment search tools like blast[14]. The consideration of only sequence similarity often leads to the exclusion of proteins that diverge in sequence but share a similar function. To address this limitation, novel approaches for characterizing the functions of unknown proteins have emerged. In particular, natural language processing (NLP) methods have gained traction. These techniques exploit the capability of representing amino acids in protein sequences through one-hot encoding with a single letter. By training

these models, numerical vectors can be assigned to encapsulate various protein characteristics ranging from sequence to structure and function. With these approaches, ideally, proteins with similar functions exhibit similar vectors, and those vectors with proper labeling can be used for classification tasks. Therefore, in such a scenario, unknown PET degrading proteins can be inferred using a classification model trained on a custom build dataset.

1.3.2 Engineering of hydrolase for PET depolymerization

The advent of genetic engineering offers unprecedented opportunities for the precise manipulation of biological systems. With this new technology, researchers can engineer organisms with tailored functionalities, unlocking novel pathways to address environmental challenges. In the future, the convergence of genetic engineering with other cutting-edge technologies, such as synthetic biology and machine learning, holds immense promise for accelerating progress in plastic degradation. Using interdisciplinary approaches from different scientific disciplines, the development of new plastic-degrading proteins will be faster and more efficient. An example of using a hybrid approach for the development of an engineered protein is MutCompute[15]. MutCompute employs an algorithm that understands the specific chemical surroundings of amino acids by utilizing a self-supervised 3D convolutional neural network (CNN) trained on a dataset of 19000 protein structures sourced from the Protein Data Bank (PDB). Essentially, it predicts the locations within a protein where wild-type amino acids are not optimized for their immediate surroundings. It can be used to confront wild-type proteins with modified ones to identify positions where wild-type amino acid residues fit less well than potential substitutions. An example of MutCompute application is the Hongyuan Lu et al. paper[16] where they tried to perform an enigneering of a hydrolase for PET depolymerization. In that paper, they found four mutations that resulted in the highest improvements, both singly and in combination for the PET degradation. Using these four mutations across three PETase the researchers were able to produce a mutated enzyme called FAST-PETase that emerged as an excellent canditate for PET degradation.

1.4 Aim of the work

The central intuition of this research is based on the fact that microorganisms that live in a plastic-contaminated environment are subject to evolutionary pressure. This claim stems from the emergence of modern materials, such as plastic-based products, that have become ubiq-

uitous in contemporary society, but were absent or scarce during the evolutionary history of these microorganisms. Unlike natural organic matter, plastics possess unique chemical compositions and degradation profiles, making them difficult to metabolize by traditional microbial degradation processes [17], consequently microorganisms residing in landfills encounter novel selective pressures imposed by the influx of plastic-derived nutrients. Furthermore, the rapid proliferation of plastic-based products in recent decades intensifies the magnitude of this evolutionary pressure[8] since the global environment is faced with an unprecedented abundance of these synthetic materials.

Evolutionary pressure leads to the adaptation of microorganisms that inhabit such specific environments. Due to the high levels of plastic contamination in these habitats, microorganisms are pushed to evolve proteins that were originally meant to degrade biological substances, such as a cutin, into proteins capable of breaking down synthetic materials such as plastic. Analyzing the sequence and structure of proteins is crucial to understanding their functionality. These characteristics can be utilized to predict the function of an unclassified protein by comparing its sequence and structural features to those of already classified proteins. Given the very large number of proteins that need to be analyzed to identify the common sequence and structural characteristics linked to a particular function, a computational method is essential to detect these patterns.

The objective of this thesis work is to develop a classification model capable of predicting promising PET degradation proteins. The microorganisms from which these proteins can be extracted may have evolved in environments such as landfills with high concentrations of plastic products. Given that scenario, we think they can be hypothesized to have evolved to utilize plastic as an additional carbon source. The aim of the work is to identify and characterize these proteins, offering insight into bioengineering solutions for plastic waste management.

The main objective is thus to develop a machine learning model for classification using a dataset that includes proteins related to PET degradation and proteins that may or may not be related to it. By including proteins from both categories, the model can learn to distinguish between features characteristic of PET degradation proteins and those that are not. Additionally, given the strict relationship between protein sequence, structure and function, this work explores the information value of the former two in predicting the latter. This approach enables the identification of key molecular signatures or patterns indicative of PET degradation capability, facilitating the prediction of unknown promising PET degradation proteins.

2

Data and Methods

As introduced in the previous chapter, the idea is to build a classification model capable of identifying proteins that can perform PET degradation. The diagram in Figure 2.1 illustrates the flow of the process that aims to identify distinctive features linked to PET degradation. The objective is to extract these features in a numerical format. As depicted in the initial step of Figure 2.1, a crucial stage involves converting the proteins into numerical vectors to facilitate the training of a classification model. To that end, as we will see in the next Sections, we tested three approaches :

1. **Sequence-based approach** : Since proteins are composed of 20 different amino acids that can be encoded by single letters, a natural language model can be built to produce a numerical representation for each protein.

2. **Structure-based approach** : Using coordinates of the atoms that compose a protein, we can build a graph for each one, and through a graph embedding model transform each graph into a second numerical representation.

3. **Combined approach** : By concatenating the vector representations coming from the sequence and structure approach, we can obtain a numerical protein representation that takes into account both types of information.

Then, after proper labeling of those vectors, the idea is to train a classification model able to predict if proteins with unknown function can be associated to PET degradation.

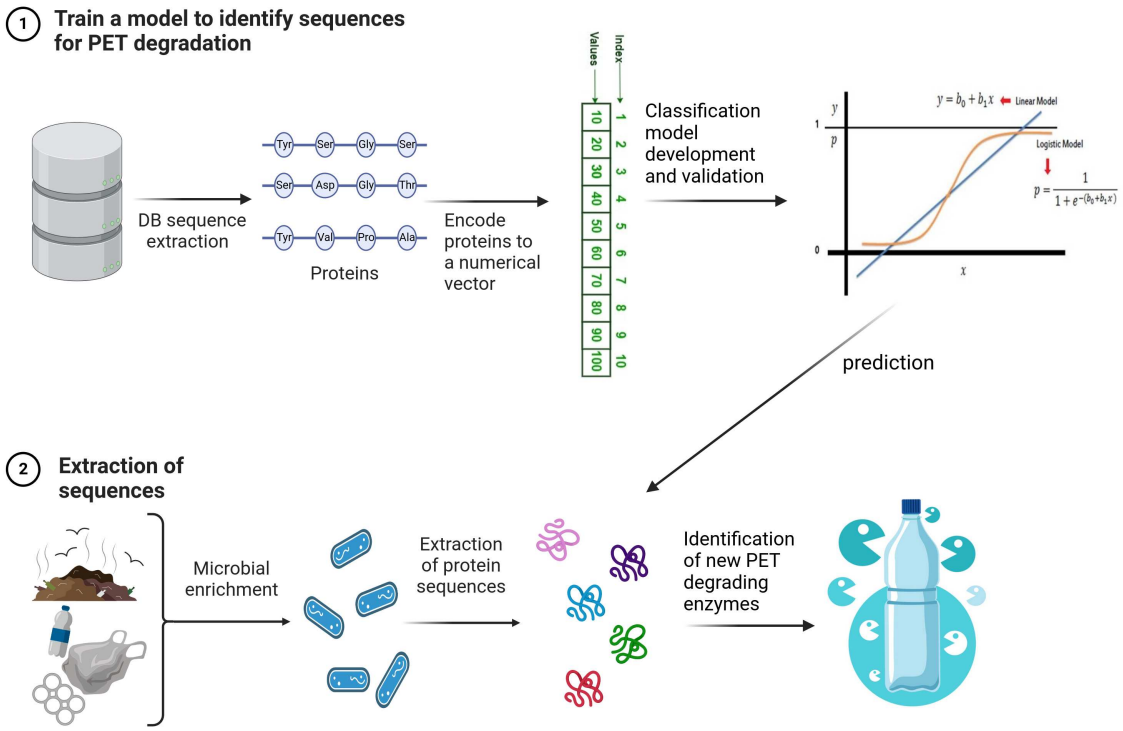

Figure 2.1: The first step is to build a dataset that contains either PET degradation proteins and either non PET degradation proteins, then from the amino acid sequence use a model that assign to each protein a numerical vector. Using these vectors train a classification model able to distinguish PET degradation proteins from the others. Finally use that model to predict which of the landfill extracted sequences can be associated to PET degradation

2.1 Data

The main data used for this thesis work were obtained from three different databases :

1. The proteins associated to PET degradation were obtained from a database named **PlasticDB**[18] which is manually curated and contains most of the known proteins that can be associated to plastic degradation. Since in this work we are interested on working on PET degradation we have selected only those related with this type of plastic. At the time of the data extraction (December 2023), 73 PET degrading proteins were available.

2. Generic proteins were obtained from **UniProt**[13], which is a comprehensive database for protein sequence and functional information. It is maintained by a collaboration between the European Bioinformatics Institute (EBI), the Swiss Institute of Bioinformatics (SIB) and the Protein Information Resource (PIR). UniProt provides a centralized repository of protein sequences, annotations, and curated information on proteins from various organisms. In this work, we have used proteins from the the UniProt/Swiss-Prot section since it contains high-quality manually curated and annotated proteins with information on sequence, function, and structure. During the work period (December 2023), in the SwissProt section 571,282 annotated proteins were present.

3. The atomic coordinates that compose a protein are encapsulate in Protein Data Bank (PDB) file. A PDB file is a standard file format used to store the three-dimensional coordinates of atoms in a molecule or complex molecular structure. These files are obtained from the **AlphaFold** database[19] and in that work are used to extract structural information from proteins.

Given the extremely high number of protein entries in the SwissProt Database, we randomly under-sampled 10,000 proteins. In this way, computational costs were kept reasonable for the available resources while drawing information from a representative protein set. Therefore, finally we obtained a dataset composed of 10,000 functionally variable proteins and 73 PET-degrading proteins. For each of these proteins, the information that we collected includes : the entry identifier, the amino acidic sequence and the EC number. Specifically :

- In the two databases we utilized, each entry as a unique alphanumerical **identification key**. However, the nomenclature for the entries differs between the databases. To prevent hidden duplicate proteins, we excluded proteins that share the same amino acid sequence. In the cases of clones between the two databases, we retained the protein sourced from the PlasticDB database. After that filtration, 9,983 proteins remained.

- **The amino acid sequence** refers to the specific order in which amino acids are arranged within a protein or peptide molecule. Proteins are composed of long chains of amino acids linked together by peptide bonds. There are 20 standard amino acids that can be found in proteins, each with its own unique chemical structure and properties. This sequence is critical for the protein's structure and function, as it dictates how the protein will fold into its three-dimensional shape and how it will interact with other molecules in the cell.

- **The Enzyme Commission (EC)** number is a hierarchical classification system for categorizing proteins based on the type of reaction they catalyze or in which are involved. Each protein is given a distinct EC number, which consists of a series of four numbers separated by periods. These numbers cover specific information about protein function and the type of chemical reaction that it facilitates.
  The structure of an EC number is as follow :

  $$\text{EC number} = a.b.c.d.$$

  where *a,b,c,d* are the classes of the EC number.

  - **Class a** : the first digit (a) refers to the type of reaction in which the protein is involved. For example, that digit can refer to a hydrolase which is a type of enzyme that catalyzes the hydrolysis reaction, involving the cleavage of chemical bonds through the addition of a water molecule.
  - **Subclass b** : the second digit (b) provides more specific information about the type of reaction. This digit further refines the classification, helping to distinguish between different types of reactions that enzymes catalyze within the same enzyme class, for example, given class a = 2 that is related to transferases, then subclass b can be :
    1 : Transferases transferring one-carbon groups.
    3 : Acyltransferases.
    4 : Glycosyltransferases.
    6 : Transferases transferring nitrogenous groups.
  - **Subsubclass c :** the third digit (c) provides more detailed information on the substrate or the chemical group involved in the reaction. This level of classification allows for a finer distinction between enzymes that catalyze similar reactions but act on different substrates or chemical groups.
  - **Serial number d :** The fourth digit (d) uniquely identifying proteins within each subclass and subsubclass, facilitating precise classification and organization of protein data.

2.2 AlphaFold2

The accurate prediction of protein structures from amino acid sequences has long been a major challenge in computational biology, with implications that range from a fundamental understanding of biological processes to drug discovery and design.
Prediction of protein structures is crucial to discovering the mechanism underlying biological functions.Experimental methods such as X-ray crystallography and cryoelectron microscopy have traditionally been used to determine protein structures, but these approaches are often time consuming, labor intensive, and costly. AlphaFold2[1] outperforms its predecessor, demonstrating groundbreaking performance in the Critical Assessment of Structure Prediction (CASP) competition, defining that model as the state of the art of protein structure prediction from sequence. The general scheme on how AlphaFold2 works can be visualized in that image2.2 and can be divided into 3 main blocks :

1. First of all AlphaFold2 query the input amino acid sequence through different databases of protein sequences and constructs a multiple sequence alignment (MSA). Multiple Sequence Alignment (MSA) is a bioinformatic technique used to align three or more biological sequences (such as DNA, RNA, or protein sequences) to identify similarity regions that may be functional, structural, or evolutionary related. This enables us to determine the parts of the sequence that tend to mutate more and enables us to detect the correlations between them. AlphaFold2 also tries to identify proteins that may have a structure similar to the input (*template*), and constructs an embryonic representation of the structure called "*pair representation*".This is, in essence, a model in which amino acids are likely to be in contact with each other.

2. In the second part of the image2.2, Alphafold2 takes the alignment of multiple sequences and templates and passes them through a *transformer* (that is an architecture that we will see in the next Section 2.3.1) that can identify which pieces of information are more informative. The objective of this part is to refine both the representations of the MSA and pair interactions and also exchange information between them.

3. In the last part of the diagram, both the refined MSA representation and the pair representation are leveraged through another neural network in order to construct a three-dimensional structure of the input protein sequence.

One last piece is that the model works iteratively, after generating the final structure, it will take all the information (MSA representation, pair representation, and predicted structure) and pass it back to the beginning of the Evoformer blocks2.2, this allows the model to refine its predictions. After that overview on how AlphaFold2 works, we will go deeper into the details.

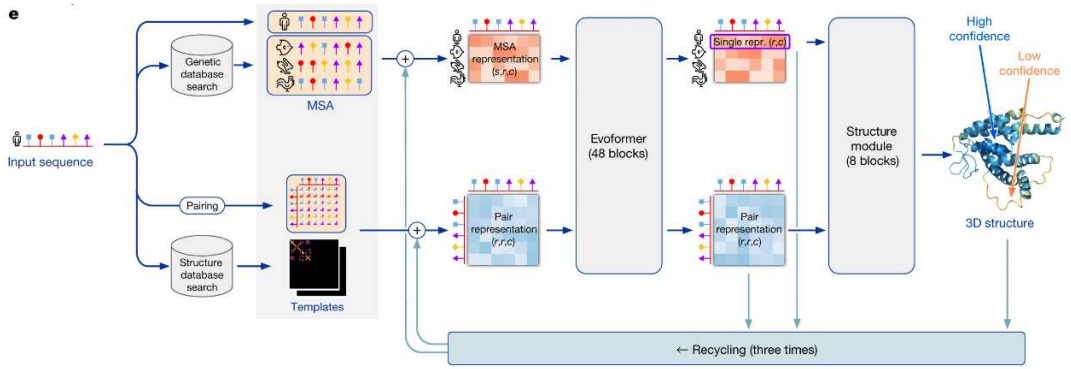

Figure 2.2: Diagram of AlphaFold2 published in the official Nature paper[1].

Preprocessing :

AlphaFold2 is equipped with a pipeline to query different databases to produce an MSA and a list of templates from an input sequence. In Multiple Sequence Alignment (MSA) the amino acid sequence is compared across a large database (like UniProt[13]) producing an output with all the aligned sequences that share a portion with the input sequence given a threshold, e.g., in the output consider a protein only if it shares at least a threshold value of sequence identity, that value can be, for example, 70%. The underlying idea on why MSA is used in AlphaFold2 is that : when two amino acids are in close proximity, mutations in one of them are likely to be promptly followed by mutations in the other in order to preserve the structure, this phenomenon is called coevolution. An example of coevolution can be the following one : When a protein contains an amino acid with a negative charge (such as glutamate) in close proximity to an amino acid with a positive charge (such as lysine), the interaction between them can play a crucial role in determining the protein's structure. Thus, if the first amino acid changes to one with a positive charge, the second amino acid will likely undergo a mutation to acquire a negative charge. This evolutionary pressure is necessary for proper protein folding; otherwise, the protein may lose its function. A visual explanation of this phenomenon is illustrated in the Figure 2.3. Finding templates follows a different but closely related principle. The idea behind template construction is that proteins tend to mutate and evolve but their structure tends to remain similar despite changes, we can see that for example in the image 2.4 where four different types of myoglobin proteins correspond to 4 different organisms are displayed. Although these proteins appear quite similar, some pairs of proteins exhibit unexpectedly low sequence similarity. For example, the protein in the lower right corner only shares approximately 25% of its amino acids with the protein in the upper left corner. In most cases, however, conserva-

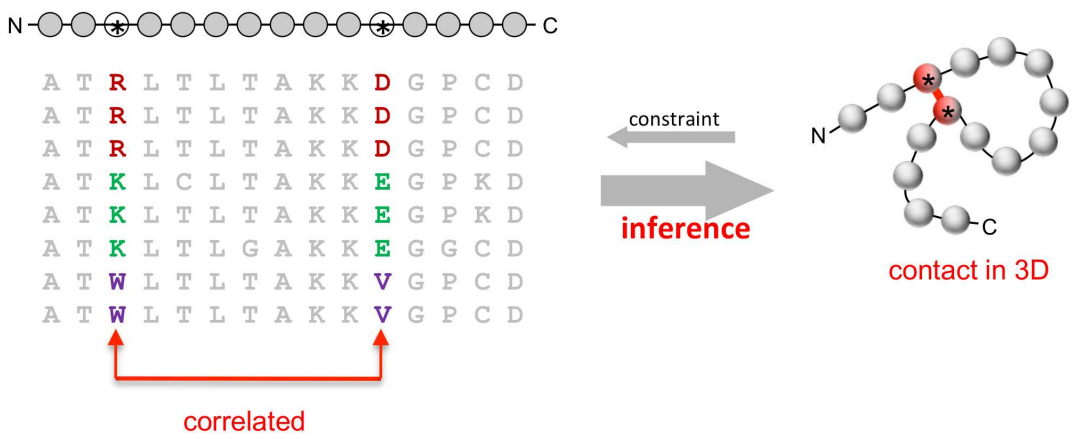

Figure 2.3: Schematic of how co-evolution extract information about protein structure from a MSA. Image taken from Marks DS et al. [2]

tion occurs on a smaller scale, where pieces of the protein(like the active center of an enzyme) remain unchanged while their surrounding evolve. Using the right method It is possible to identify some of these conserved fragments and use them as a guide to construct the structure.

The Evoformer (evolutionary transformer) model :

The task of the Evoformer is to extract information from the multiple-sequence alignment and the templates. The central behind idea of the Evoformer is that the information flows back and forth through the network, at every cycle the model leverages the current structural hypothesis to improve the assessment of the multiple sequence alignment, which turns to a new structural hypothesis. The evoformer architecture uses two transformers[20], each head is specialized for the particular type of data it is looking at, either a multiple sequence alignment or a representation of pairwise interaction between amino acids. The two transformers are as follows :

1. The **MSA transformer** computes attention over a large matrix of protein letters. To reduce what would otherwise be an impossible computational cost, the attention is factorized in the 'row-wise' and 'column-wise' components. That is, the network first computes attention in the horizontal direction, allowing the network to identify which pairs of amino acids are more related; and then in the vertical direction, determining which sequences are more informative. The most important feature of the AlphaFold2 MSA transformer is that the row-wise (horizontal) attention mechanism incorporates information from the 'pair representation'. When computing attention, the network adds a bias term that is calculated directly from the current pair representation. This trick augments the attention mechanism and allows it to pinpoint interacting pairs of residues.

2. The other transformer that acts on the pair representation works in a similar manner, but focusing on different details. The key feature of this network is that the attention is

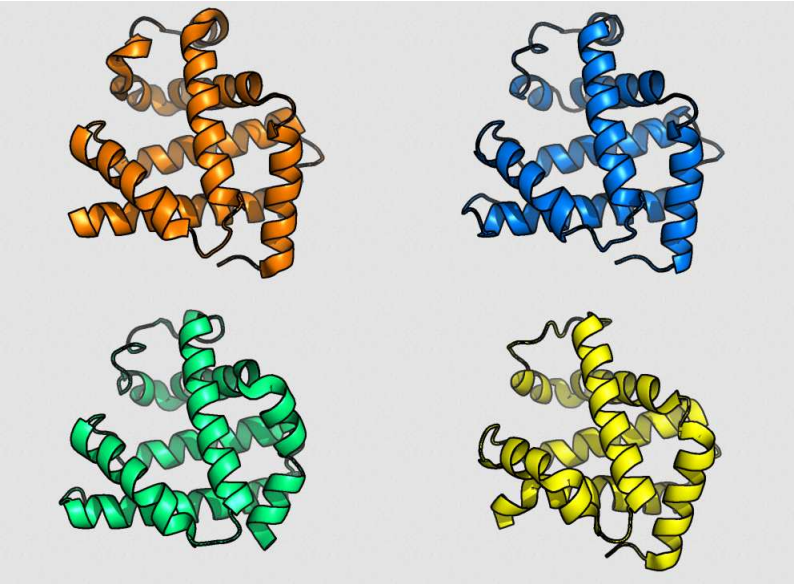

Figure 2.4: Protein structures of human myoglobin (top left), african elephant myoglobin (top right, 80% sequence identity with human myoglobin), blackfin tuna myoglobin (bottom right, 45% sequence identity with human myoglobin) and pigeon myoglobin (bottom left, 25% sequence identity with human myoglobin)

arranged in terms of triangle of residues, the intuition is to enforce the triangle inequality. The triangle inequality is a fundamental concept which states that in a metric space, the distance between two points is always less than or equal to the sum of the distance between those two point and a third point. Formally given 3 points A, B, C in an euclidean space the triangle inequality can be expressed as :

$$d(A, B) \leq d(A, C) + d(C, B) \tag{2.1}$$

After 48 iterations the network has built a model of the interactions within the proteins.

The structure model Until now the model has generated two representations :

- A representation of the multiple sequence alignment (MSA) which captures sequence variation.
- Representation of the pair residues, which captures which residues are likely to interact with each other.

In the structure model every amino acid is modelled as a triangle (as we can see in the Figure 2.5) representing the three atoms of the backbone. These triangles float around in space and are moved by the network to form the structure. These transformations are parameterised

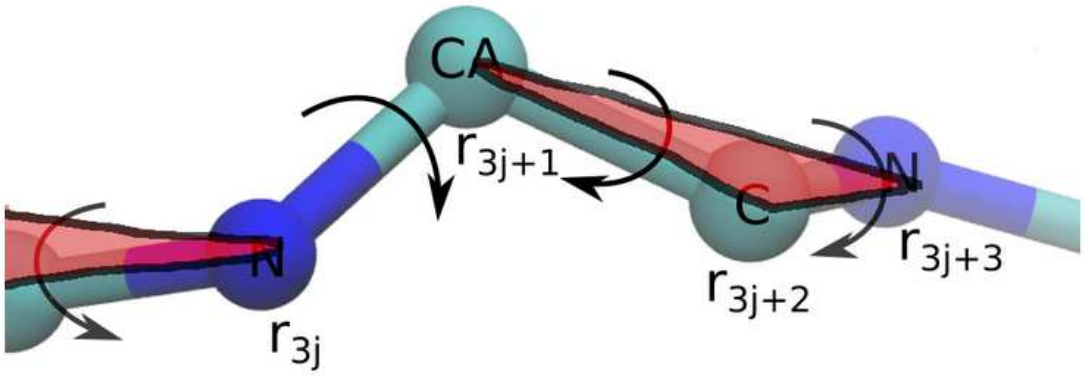

Figure 2.5: Graphical representation on which is visualized how each aminoacid is modeled as a triangle, the triangle vertex are : carbon-alpha, amine group and carboxylic group. Those elements are the fundamental building block of each aminoacid.Image taken from the Open Fold 2 webpage

as affine matrices, which are a mathematical way to represent translations and rotations in a single 4x4 matrix. In the initial phase of the structure module, the residues are initially positioned at the origin of the coordinates. During each iteration, AlphaFold 2 generates a series of affine matrices to translate and rotate the residues in space. This representation does not reflect any physical or geometrical assumptions and, as a result, the network has a tendency to generate structural violations. Since any rotation or translation of the data yields the same result, a newly attention mechanism called "Invariant Point Attention" (IPA) is introduced. IPA is invariant to rotations and translation, and requires significantly less data to discern inaccurate models, consequently enhancing its learning capacity. Finally, after multiple iterations the model obtains the protein structure prediction.

2.3 From sequence to vector

For converting an amino acids sequence into a vector, the idea is to use a natural language processing (NLP) model. We used that type of model because the 20 possible different amino acids can be encoded by a single letter and proteins can be seen as pieces of text and converted by a NLP model into a vector. In this Section, we present and provide an explanation of the underlying intuition behind the chosen model.

2.3.1 BERT

BERT (Bidirectional Encoder Representations from Transformers) is a state-of-the-art NLP model introduced in 2018 [21]. BERT is based on the transformer architecture, which is a type of artificial neural network utilizing self-attention mechanisms to effectively capture contextual information from both left and right contexts in a sentence.

One of the key innovations of BERT is its pre-training strategy, where the model is first trained on large amounts of unlabeled text data in an unsupervised manner. During this pre-training phase, BERT learns to predict missing words in sentences based on the surrounding context. This process enables BERT to capture rich semantic representations of words and sentences, which can then be fine-tuned for downstream NLP tasks such as text classification and question answering. BERT's bidirectional nature allows it to understand the context of a word by considering all of its surrounding words, leading to more accurate representations of word. This is done through the self-attention step that will be described below. Additionally, BERT can be fine-tuned on specific tasks with relatively small amounts of labeled data, making it highly versatile and applicable to a wide range of NLP task.

In the following, we present the main steps that comprise the BERT model.

Tokenization :

In the first step, tokenization, the words are converted into numerical vectors. In the case of amino acid sequences, where each amino acid is represented by a one-hot encoding letter, this process involves converting each letter into a numerical vector or embedding representation. This numerical representation allows the model to process and manipulate the input data mathematically, facilitating further analysis and processing by subsequent layers in the neural network. Returning to the more intuitive analogy of a sentence, the tokenization process comprises distinct elements that are summarized creating a unique vector for each word. These elements are as follows :

- **token embeddings** : text is subdivided into tokens i.e. uniform blocks that make up the sentence.
- **sequence embeddings** : codify each sentence as a unique vector.
- **position embeddings** : codify the position of the words in the sentences. This is an important information since intuitively the position of word in a sentence can change its meanings. The positional encoding of each word is calculated as follows :

$$PE_{pos,2i} = sin(pos/10000^{\frac{2i}{d_{model}}})$$

$$PE_{pos,2i+1} = cos(pos/10000^{\frac{2i}{d_{model}}}),$$

where pos is the position of each word and i is used to map the column indices of the embeddings.

Figure 2.6 shows an example of tokenization of two sentences.

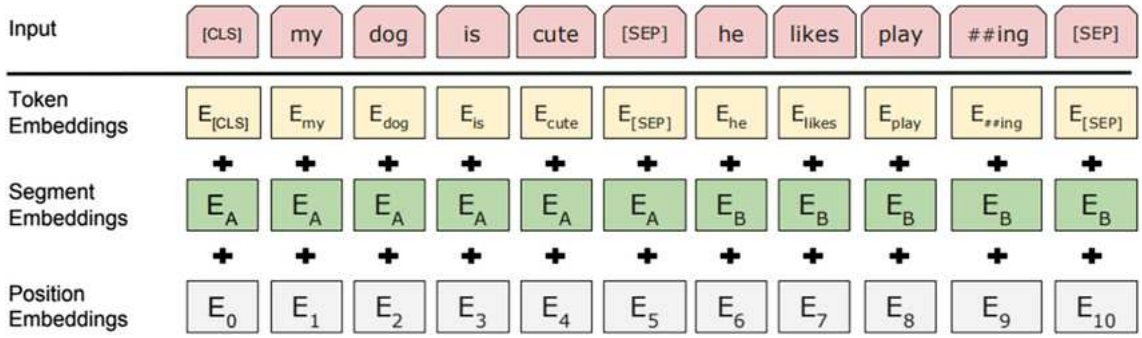

Figure 2.6: Input embeddings are the sum of the token embeddings, the segmentation embeddings, and the position embeddings. Image taken from this paper [3]

Masking :

Normally, approximately 15% of the input elements are masked, and the goal is to predict these masked elements to optimize a log-likelihood function which is expressed as follows :

$$L(X, \theta) = \mathop{\mathbb{E}}_{x \sim X mask} \mathbb{E} \sum_{i \in mask} logp(x_i | x_{j \notin mask, \theta}) \tag{2.2}$$

Where X correspond to all input elements and $mask$ contain the indices of masked elements.

Computing of the self attention similarity :

Three matrices, identified as Q, K, and V, are obtained as follows :

$$Q = X \cdot W_Q$$

$$K = X \cdot W_K$$

$$V = X \cdot W_V$$

W_Q, W_K, W_V are the parameters learned through backpropagation, and X are the input vectors of our model. $Q, K, V \in \mathbb{R}^{n \times m}$ where n is equal to the number of elements in input, so in the case of a protein, for example, it will be equal to the number of amino acids that compose

a protein and m is the dimension of the embedding. Finally, a $n \times n$ similarity matrix that we call S is obtained through the following matrix operation :

$$S = softmax(\frac{QK^T}{\sqrt{d}}) \tag{2.3}$$

In order to obtain a similarity matrix, the model use the cosine similarity. Cosine similarity is a measure of similarity between two non-zero vectors. It corresponds to the cosine of the angle between the vectors, which is the dot product of the vectors divided by the product of their lengths. The cosine similarity belongs to the interval $[-1, 1]$. For example, two proportional vectors have a cosine similarity value of 1, two orthogonal vectors have similarity of 0 and two opposite vectors have similarity of -1. Given two vectors A and B the cosine similarity is mathematically defined as :

$$S_C(A, B) := cos(\theta) = \frac{A \cdot B}{\|A\| \, \|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}} \tag{2.4}$$

After the similarity matrix is obtained the softmax is applied on it. Softmax function takes as input a vector $z \in \mathbb{R}^K$ and normalized it into a probability distribution consisting of K probabilities proportional of the input numbers. Mathematically, is defined as :

$$softmax(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{2.5}$$

$$i = 1, .., n \; and \; z = (z_1, ..., z_k) \in \mathbb{R}^k$$

The softmax function is applied to the resulting $n \times n$ matrix 2.3, providing a probability representation named similarity matrix which indicate the correlation between elements. This considers their position and meaning within their specific context. After softmax operation, the resulting similarity matrix is multiplied by the V matrix, resulting in a matrix with a shape of $n \times m$, referred to as the attention value matrix. Each row of this matrix represents the embedding of the input sequence, where in the case of text, it corresponds to the embedding of each word and, in the case of a protein, to the embedding of each amino acid. A general scheme on how the attention matrix of the sentence "*I am a student*" is obtained can be seen in Figure 2.7. The matrices Q and K are updated with backpropagation in order to find a good embedding of input elements and obtain a good separation within elements that have distant similarity values.

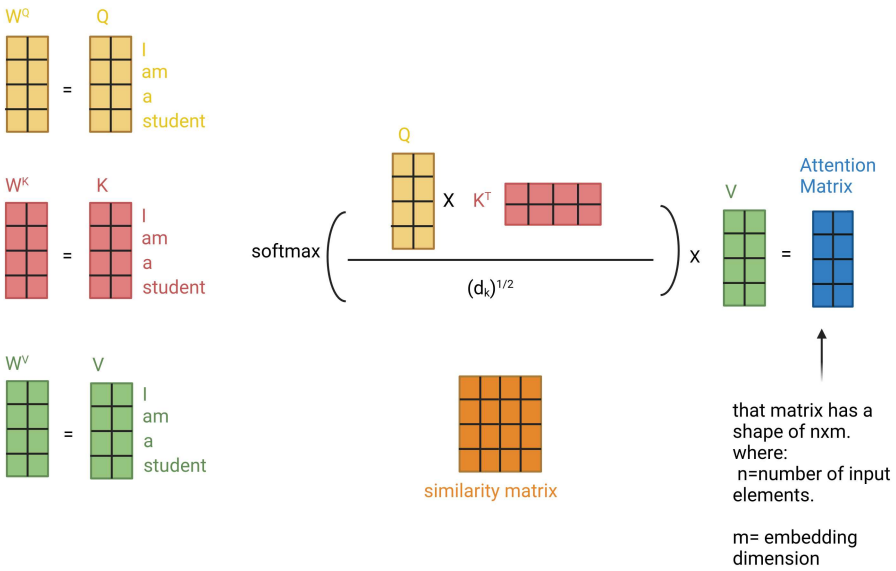

Figure 2.7: Computation of the attention value matrix for the sentence *"I am a student"*.

Intuition on how the attention mechanism works :

As we can deduce from the previous Section, the attention mechanism is a key operation in the BERT model. Essentially, its main concept revolves around assigning a weight to each element of the input sequence based on its context. This is achieved through a weighted combination of all other elements within the input. If we consider the input as a sentence, we can imagine that each word is encoded as a vector (referred to as the word's embedding). Using these embeddings, we can generate an $n \times n$ matrix where n represents the number of words. In this matrix, each position corresponds to the dot product between one word and another word in the sentence. This matrix indicates the degree of correlation between each word and all the others in the sentence. For example, the initial words in the sentence will possess a similarity value to all the others, which can then be used to determine the new positioning of that word within the embedding space. To give a better idea on how the attention mechanism works we provide an example through the use of the two following sentences :

"an apple and an orange"

"an apple phone"

As we can imagine the word "apple" in the two sentences given the context has a different meaning, using the attention mechanism we will show which that word can be discriminated based on the context. For simplicity in that example, we assume that the embedding space is $\mathbb{R}^3$, the x axis is related to the technology of a word, the y axis is related to the fruitiness, and the z axis is related to others do not specified characteristics. The initial coordinates of the embedding are in the following Table 2.1. A detail that we can notice is that the initial embedding of the word *apple* has coordinate of $[2, 2, 0]$ this points out his double meaning, later we will see how the assigned embedding of the word *"apple"* change based on context. Starting from the initial

	Technology	**Fruitiness**	**Other**
orange	0	3	0
phone	4	0	0
apple	2	2	0
and	0	0	2
an	0	0	3

Table 2.1: Coordinate of words embedding

embedding of the sentence "*an apple and an orange*" we compute a similarity matrix using cosine similarity. Following this, we apply the softmax function in order to obtain the resulting matrix 2.2. From the similarity matrix referenced as 2.2, we observe that the similarity between

	orange	apple	and	an
orange	0.57	0.43	0	0
apple	0.43	0.57	0	0
and	0	0	0.5	0.5
an	0	0	0.5	0.5

Table 2.2: Table similarity

the words 'apple' and 'orange' is 0.43. Intuitively, this implies that the initial embedding vector of the word 'apple' with coordinates [2,2,0] will be adjusted towards the embedding vector of 'orange' [0,3,0] by 0.43%, resulting in the new embedding coordinates for the word 'apple' as $[1.14, 2.43, 0]$. This makes sense since in that context the word apple is more related to fruitiness with respect to technology.

If we compute with the same logic a table similarity of the other sentence "*an apple phone*" the resulting embedding vector of the word 'apple' will be more shifted towards the embedding of the word phone. This because intuitively in that context 'apple' is more related to technology with respect to fruitiness.
In conclusion, the attention mechanism enables the creation of embeddings where words sharing similar contexts are positioned closely together in the embedding space.

2.3.2 ESM1b : a BERT-based model

ESM1b is a transformer protein language model based on BERT architecture (Section 2.3.1), trained on protein sequence data without label supervision, developed by Rives et al. [22].
The model is pre-trained on Uniref50[13] with an unsupervised masked language modeling (MLM) objective, meaning that the model is trained to predict amino acids from the surrounding sequence context. This pre-training objective allows ESM1b to learn generally useful features that can be transferred to downstream prediction tasks. UniRef50 is a database containing approximately 30 million protein sequences maintained by the UniProt Consortium[13], which aims to provide clustered sets of protein sequences at a 50% sequence identity, e.g., set that contains sequences that share at least 50% similarity in their amino acid sequence. This clustering approach aims to minimize redundancy in the database. Training of that model is obtained with the objective of modeling in masked language, randomly masking 15% of the amino acids in the input sequence. For each protein, the model produces a numeric vector with 1280 elements. The main advantage of using a NLP models like ESM1b is that they can infer information about the structure and function of proteins without further supervision, i.e. they are capable of zero-shot transfer to structure and function prediction, since the idea is which the biological function and structure are recorded in the statistics of protein sequences selected through evolution in millions of years.
In the specific case of ESM1b, thanks to self attention layers the model can be interpreted as way to consider all possible interactions between amino acid pairs. This is very valuable since the proteins fold in a 3D structure, so amino acids that are distant in sequence can be near in the space, for example, forming interactions that are very important for functional domains or structure.

2.4 From graph to vector

In this Section, we present the different graph embedding methods used in this work. Graph embedding techniques enable us to convert complex graph structures into continuous vector spaces, facilitating the application on them of traditional machine learning algorithms. These vectors encapsulate meaningful information about the nodes, edges, and overall topology of the graph.

2.4.1 Graph2vec

Problem statement :
Given a set of graphs $G_1, G_2, ..., G_N$, the Graph2Vec[23] model intends to learn $\delta-dimensional$ distributed representations for every graph $G_i \in G$. The matrix representations of all graphs is denoted as $\phi \in \mathrm{R}^{|G|\times\delta}$ where $|G|$ corresponds to the number of graphs and δ corresponds to the dimension of each embedding. In particular, let $G = (N, E, \delta)$, represent a graph where N is a set of nodes, $E \subseteq (N \times N)$ be a set of edges, and λ is a function that maps the nodes to a unique label $\lambda : N \to l$ if the graph G is labeled.
The goal of graph2vec is to learn an embedding (a numerical representation) of graphs using the recently proposed embedding techniques in NLP.

Background : Skipgram Word and document embedding models :
New neural embedding methods such as word2vec[24] use a simple and efficient feed-forward neural network called skipgram to learn embedding representations of words.
word2vec works based on that intuition :

words that appear in similar contexts tend to have similar meanings and hence should have similar vector representations.

To achieved that word2vec model try to learn a target word representation given a context defined as fixed number of words surrounding it. Given a sequence of words $\{w_1, w_2, ..., w_t, ..., w_T\}$, the target word w_t has to be learned given a context window of size c, the objective is to minimize the following log-likelihood :

$$\sum_{t=1}^{T} logPr(w_{t-c}, ..., w_{t+c}|w_t) \tag{2.6}$$

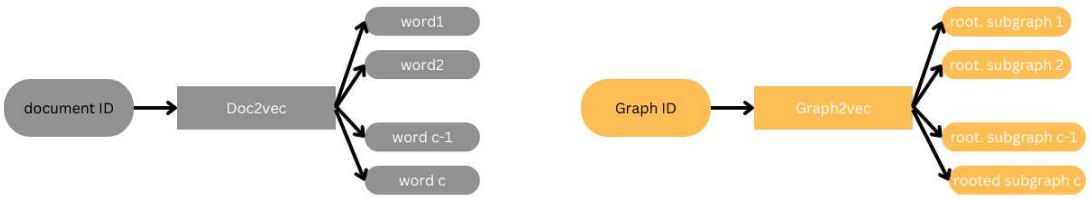

Figure 2.8: Doc2vec and Graph2vec analogy,basically different subgraphs compose graphs in a similar way that different words compose sentence or document.

where $w_{t-c}, ..., w_{t+c}$ are the context of the target word w_t. The probability $Pr(w_{t-c}, ..., w_{t+c}|w_t)$ is computed as :

$$Pr(w_{t-c}, ..., w_{t+c}|w_t) = \prod_{-c \leq j \leq c, j \neq 0} Pr(w_{t+j}|w_t) \tag{2.7}$$

Since the context words and the target word are assumed to be independent $Pr(w_{t+j}|w_t)$ is defined as :

$$Pr(w_{t+j}|w_t) = \frac{exp(\vec{w} \cdot \vec{w'}_{t+j})}{\sum_{w \in V} exp(\vec{w} \cdot \vec{w})} \tag{2.8}$$

where $\vec{w}$ and $\vec{w'}$ are the input and output vectors of word w and V is the vocabulary of all the words. The posterior probability in equation 2.7 is obtained through negative sampling. This implies selecting a small subset of words at random that are not in the target (w_t) context and updates their embeddings in every iteration instead of considering all words in the vocabulary. So if a word w appears in the context of another word w' then the vector embedding of w is closer to w' compared to any other random word in the vocabulary. Recently, doc2vec which is a straightforward extension to word2vec is introduced, basically it is able to learning embedding representation of arbitrary length word sequences such as sentences, paragraphs and even whole large documents.

Intuition of Graph2Vec :

The idea presented before on word and document embedding can be extended to graphs. We can imagine a graph as a document and the rooted subgraphs surrounding every node in the graph as words that compose the document. So, basically, a graph G is divided into different rooted subgraphs with a fixed number of neighbors within a certain degree. We can see an analogy of the Doc2vec and Graph2Vec methods in the following Figure 2.8.

Extraction of Rooted Subgraphs :

Let H be a non-negative integer parameter which defines the maximum height of rooted subgraphs. For every node *v* in a graph G, Graph2vec generated (H+1) rooted subgraphs whose roots are *v*. For $0 \leq t \leq H$, the t-th subgraph rooted at *v* describes the surrounding nodes within the *t* hops. After all, if G consists of *n* nodes, Graph2vec creates $n(H+1)$ rooted subgraphs.

Learning Embedding of entire graphs :
After extracting the rooted subgraphs of a graph G, Graph2vec uses the skip-gram model 2.4.1 to learn the embedding of the graph.
Given a set of graphs $G_1, G_2, ..., G_N$ and their subgraphs $c(G_1), c(G_2), ..., c(G_N)$, Graph2vec learns $\delta-$dimensional embedding $f(G_i)$ for G_i and $\delta-$dimensional embedding for each member subgraph in $c(G_i)$. The model considers the probability that the j-th subgraph sg_j in $c(G_i)$ occurs in G_i and maximize the following log-likelihood :

$$\sum_{j=1}^{n_i(H+1)} logPr(sg_j|G_i) \tag{2.9}$$

Where n_i denotes the number of nodes in G_i, and the probability $Pr(sg_j|G_i)$ is defined as :

$$Pr(sg_j|G_i) = \frac{exp(f(G_i) \cdot f(sg_j))}{\sum_{sg \in V_{OC}} exp(f(G_i) \cdot f(sg))} \tag{2.10}$$

Where V_{OC} denotes the vocabulary of subgraphs across all the graphs. After the training converges, graphs which share many common rooted subgraphs are mapped to similar positions in the vector space. The skip-gram model can be trained efficiently with negative sampling.

2.4.2 GL2vec

GL2vec[25] is inspired by Graph2Vec, addresses its limitations by implementing specific precautions. It overcomes Graph2Vec's inability to handle edge labels and prevent loss of structural information crucial for evaluating structural similarity. For avoiding those limitations GL2vec introduces the line graph concept.

Line Graph :
Given a graph $G = (V, E)$, its line graph $L(G) = (LV, LE)$ represents the adjacency rela-

tionship between edges in G. To construct $L(G)$, the edges of G are converted to the nodes in $L(G)$. In $L(G)$ two vertices $v(e_i)$ and $v(e_j)$ are connected by an edge if e_i and e_j share a common endpoint in G. For example, look at the Figure 2.9 since edge (v_1, v_2) and edge (v_1, v_4) share the same endpoint v_1 in G, an edge connects the node (v_1, v_2) and the node (v_1, v_4) in $L(G)$. The line graph has an attractive property that the edge features of a graph G can become the node labels in $L(G)$. Furthermore, because $L(G)$ does not remove the node labels in G, $L(G)$ is suitable to treat the structural information about G independently of the node labels in G. The GL2vec woking operations are as follows :

1. Given a set of graphs $G_1, G_2, ..., G_N$, we construct their line graphs $L(G_1), L(G_2), ..., L(G_N)$. We change the node labels in $L(G_i)$, depending on whether the graph data set has edge labels or not.
2. By applying Graph2vec to $G_1, G_2, ..., G_N$, the embedding $f(G_i)$ of each G_i is derived.
3. By applying Graph2vec to $L(G_1), L(G_2), ..., L(G_N)$, the embedding $g(L(G_i))$ of each $L(G_i)$ is derived.
4. By concatenating $f(G_i)$ to $g(L(G_i))$ the final embedding of G_i is obtained.

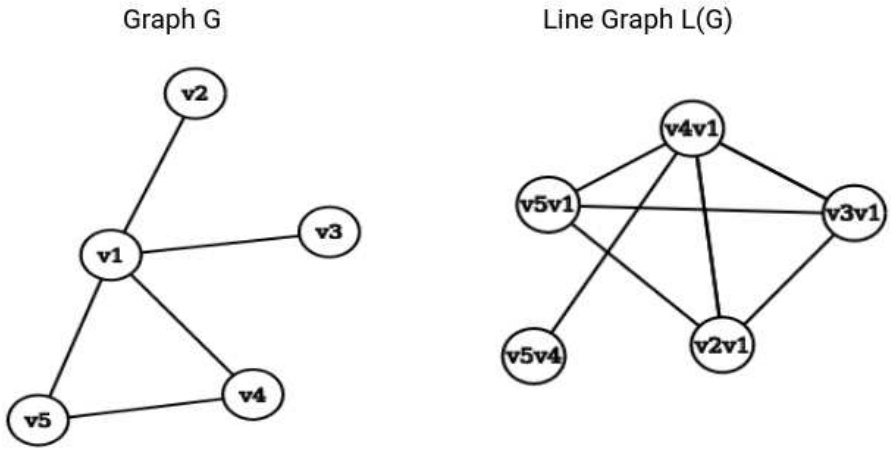

Figure 2.9: Graph G and its line graph $L(G)$.

2.4.3 FeatherGraph

The main idea of FeatherGraph[26] is to describe the distribution of node features in a neighborhood using characteristic functions obtained by random walk. The correlation of attributes

is known to decrease with the decrease in tie strength and with increasing distance between nodes. FeatherGraph uses a random-walk based to tie strength. Where tie strength at the scale r between source and target node pair is the probability of an r-length random walk from the source node ending at the target. From that intuition we define the r-scale random walk weighted characteristic function as the characteristic function weighted by these tie strength. The goal of Feather graph algorithm is to efficiently evaluate this function for multiple features on a graph.

Node feature distribution characterization :
We assume that we have an attributed and undirected graph $G = (V, E)$. The nodes of G have feature described by the random variable X, specifically defined as the feature vector $x \in \mathbb{R}^{|V|}$. We are interested in describing the distribution of this feature in the neighborhood of $u \in V$. The characteristic function of X for the source node u can be defined as follows :

$$E[e^{i\theta X}|G, u] = \sum_{w \in V} P(w|u) \cdot e^{i\theta x_w} \tag{2.11}$$

Where the affiliation probability $P(w|u)$ describes the strength of the relationship between the source node u and the target node $w \in V$. It is important to remember that the source node u and the target nodes cannot necessarily be directly connected and $\sum_{w \in V} P(w|u) = 1$ holds $\forall u \in V$. Using the Euler identity, we can obtain the real and imaginary part of the function 2.11. Since the affiliation probability $P(w|u)$ between the source u and target w is parameterized, we introduce a parametetrization which uses random walk transition probabilities. Suppose that the neighborhood of u at scale r consists of nodes that can be reached by a random walk in r steps from the source node u. We are interested in describing the distribution of the feature in the neighborhood of $u \in V$ at scale r with the real and imaginary parts of the characteristic function, which are :

$$Re(E[e^{i\theta X}|G, u, r]) = \sum_{w \in V} P(v_{j+r} = w|v_j = u) cos(\theta x_w) \tag{2.12}$$

$$Im(E[e^{i\theta X}|G, u, r]) = \sum_{w \in V} P(v_{j+r} = w|v_j = u) sin(\theta x_w) \tag{2.13}$$

Where $P(v_{j+r} = w|v_j = u) = P(w|u)$ is the probability of a random walk starting from source node u, and reaching the target node w in the r^{th} step. From these characteristic func-

tions (one for each node), we can obtain a matrix that describes the features distribution around nodes. That representation can be seen as node embedding. For a whole graph representation those matrices are pooled with a permutation invariant aggregation function such as the mean,maximum, and minimum.

2.4.4 Wavelets-Based graph embedding

Wavelet[27] is a graph embedding method that considers node features as random variables and examines the distribution of node features in subgraphs.
Given a graph $G = (V, E, A)$ be an undirected and unweighted graph, where V is a set of vertices, $E \subseteq V \times V$ is the set of unweighted edges between vertices V and $A \in \mathbb{R}^{N \times m}$ describes the attributes of each node in the network. The goal is to represent the entire graph as one d-dimensional vector $X \in \mathbb{R}^d$. The idea is to calculate the topological similarity of the nodes based on diffusion wavelets and use that mathematical tool to capture the distribution of the node features in subgraphs. Finally, aggregating the characteristic functions of k-hop subgraphs representative points are picked and concatenated in order to get the graph-level embedding representation.

Topological Wawelet Similarity :
The Laplacian matrix L is the difference between the adjacency matrix and the degree matrix of a graph, from which it is possible to obtain an eigenvalue of the temporal frequencies of a signal on the graph. To obtain larger eigenvalues and smooth the signals, a filter kernel g_t with scaling parameter t is introduced. In that case, it used the spectral kernel $g_t = e^{-\lambda t}$. The spectral wavelet coefficient matrix Ψ is defined as :

$$\Psi = U diag(g_t(\lambda_1), ..., g_t(\lambda_N))U^T \tag{2.14}$$

For a given node v_i the element Ψ_{ij} represent how much energy comes from node v_j to node v_i, therefore each column of Ψ describe the distribution of energy from the other nodes. It has been proven that nodes with similar energy distribution patterns have similar structural roles in the network. So we can assume that the difference between wavelet distributions of two nodes represents their topological distance. The minimum difference of pair assignment (MDPA) can quickly measure the distance between two histograms, that measure can be used to calculate the distance between pair of nodes v_i and v_j, and the topological node similarity

can be defined as follows :

$$s(v_i, v_j) = e^{-MDPA(\psi_i, \psi_j)} \tag{2.15}$$

Sub-graph Feature Distribution

Assuming that the features of node v_i is a random vector $\vec{a_i} \in \mathbb{R}^m$. The distribution of features in subgraphs is used to recover the characteristic function of $\vec{a_i}$. Since the correlation between attributes is negatively related to the distance from the node, for a given node v_i , we consider the distribution of the characteristics in the k-hop subgraph $G_k(v_i)$. The characteristic function of $\vec{a_i}$ in $G_k(v_i)$ is defined as :

$$\phi_{v_i}{}^{(k)}(t) = \sum_{v_j \in G_k(v_i)} P(v_j|v_i) e^{ita_j} \tag{2.16}$$

The transition probability $P(v_j|v_i)$ is proportional to two factors : the similarity between nodes v_j and v_i and the influence of node v_i. By aggregating the characteristic function over all nodes, we can obtain a graph embedding.

The final embedding is constructed by concatenating the embedding with transition probability using normalized topological similarity and the embedding with transition probability using normalized node influence.

2.5 Semi-supervised learning : label propagation

Semi-supervised learning is a branch of machine learning that combines supervised and unsupervised learning using labeled and unlabeled data to improve model training for classification and regression tasks. Therefore, semi-supervised learning is generally employed for the same scenario on which supervised learning is used, it is distinguished by various techniques that incorporate labeled and unlabeled data into model training. Semi-supervised methods are especially useful in situations where obtaining a sufficient amount of data is prohibitively difficult or expensive, but large amounts of unlabeled data are relatively easy to acquire. In more specialized machine learning use cases, like drug discovery, genetic sequencing, or protein classification, data annotation is not only extremely time consuming but also requires very specific domain expertise. Our case fits this situation; in fact, in a real-word scenario, we have many proteins with no available labels. As shown in Figure 2.10, it is evident that the use of the canonical supervised method does not produce an optimal separation boundary between the two sets. In contrast, a semi-supervised approach, which leverages both labeled and unlabeled data, offers a

broader perspective of the complete dataset, enabling a more effective separation between the two sets.

A necessary condtion of semi-supervised learning is that the underlying marginal data distributin $p(x)$ over the input space contains information about the posterior distribution $p(y|x)$. If this hold true, one might be able to use unlabeled data to gain information about $p(x)$, and thereby about $p(y|x)$. If, on the other hand, this condition is not met, and $p(x)$ contains no information about $p(y|x)$, it is inherently impossible to improve the accuracy of predictions based on the additional unlabelled data [28]. Fortunately, the previously mentioned condition appears to be satisfied in most learning problems encountered in the real world, as is suggested by the successful application of semi-supervised learning methods in practice. However, the way in which $p(x)$ and $p(y|x)$ interact is not always the same. This has given rise to the semi-supervised learning assumptions :

1. **Smoothness assumption :** if two samples x and x' are close in the input space, their label y and y' should be the same.
2. **Low-density assumption :** the decision boundary should not pass through high-density areas in the input space.
3. **Manifold assumption :** data points on the same low-dimensional manifold should have the same label.

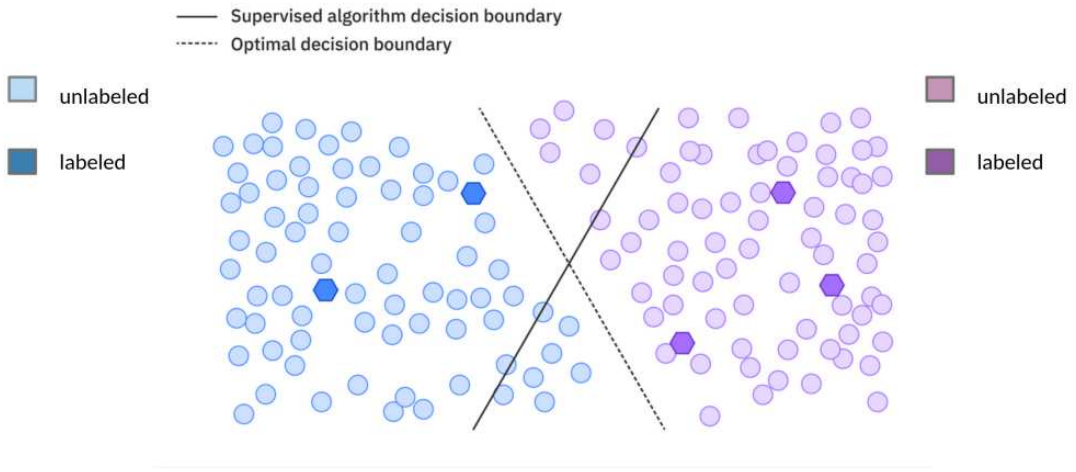

Figure 2.10: Comparison of the optimal decision boundary, accounting for both labeled and unlabeled samples, with the one obtained through supervised learning. Image taken and modified from van Engelen et al [4]

After that brief introduction on semi-supervised learning on what are the general idea behind it and why we decide to use it we start with an explanation of the classification semi-supervised learning model chosen : label propagation.

Label propagation :
The main idea of label propagation is based on the assumption that closer data points have similar labels. As a result these class labels can be propagated to the unlabeled regions, for doing that the main intuitively steps of the model are :

1. **Create a graph** : starting from the data points and considering them as nodes connecting them with edges.
2. **Determine weights** : the edges weight that connect nodes is assigned based on the distance of the nodes, the more distant the nodes are, and the less weight of the edge.
3. **perform a random walk** : for each unlabeled node perform a random walk that produces a probability distribution of the path.
4. **assign label** : based on the probability distribution previously computed assign a label to the unlabeled nodes and reiterate the random walk until the assigned label converges.

Problem formulation :
Let $(x_1, y_1), ..., (x_l, y_l)$ be labeled data where $Y_L = y_1, ..., y_l$ contains all the labels of the data. Let $(x_{l+1}, y_{l+1}), ..., (x_{l+u}, y_{l+u})$ the unlabeled data, generally $l << u$. Let $X = x_1, ..., x_{l+u}$ labeled and unlabeled data where $x_i \in \mathbb{R}^D$, the objective is to estimate Y_U (unlabeled data) from X and Y_L. Intuitively, data points that are close should have similar labels, a fully connected graph is created where nodes are all data points, both labeled and unlabeled, the edge between any nodes i, j is weighted so that the smaller the Euclidean distance is and larger w_{ij} will be. The weight between two nodes is computed as follows and is controlled by the σ parameter :

$$w_{ij} = exp(-\frac{d_{ij}^2}{\sigma^2}) = exp(-\frac{\sum_{d=1}^{D}(x_i^d - x_j^d)^2}{\sigma^2}) \tag{2.17}$$

Larger edge weights cause to a an hypothetical random walker that start from an unlabeled node to travel with more probability through them, regarding that a $(l+u) \times (l+u)$ probability transition matrix T is defined

$$T_{ij} = P(j \to i) = \frac{w_{ij}}{\sum_{k=1}^{l+u} w_{kj}} \tag{2.18}$$

Where T_{ij} is the probability to jump from from node j to i. Also a $(l+u) \times C$ label matrix is defined, where the ith row represents the label probability distribution of the node x_i, basically the algorithm stops when the rows of that matrix reach the convergence.

2.6 Dimensionality reduction : t-SNE

T-distributed stochastic neighborhood embedding (t-SNE) is a method used to visualize data from a high-dimensional space to a low-dimensional one. The method starts by converting the high-dimensional Euclidean distance between points into conditional probability. The idea is : given two points x_j and x_i if they are close in the Euclidean space the conditional probability $p_{j|i}$ will be high, otherwise it will be low. That concept can mathematically be formulated as

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-||x_i - x_k||^2/2\sigma_i^2)} \tag{2.19}$$

Where σ_i is the variance of the Gaussian distribution centered on datapoint x_i. That definition of conditional probability can cause problems when a x_i point is an outlier, and we try to map this point in the low-dimensional space. To avoid this, the joint probability p_{ij} is defined as $p_{ij} = \frac{p_{j|i}+p_{i|j}}{2n}$. In the low-dimensional space to convert distances into probabilities, a t-distribution with 1 degree of freedom is used since it has much heavier tails than a Gaussian distribution, facilitating a more effective translation from high-dimensional to lower-dimensional spaces. Using this distribution, the joint capabilities q_{ij} are defined as follows :

$$q_{ij} = \frac{(1 + ||y_i - y_j||^2)^-1}{\sum_{k \neq l}(1 + ||y_k - y_l||^2)^-1} \tag{2.20}$$

Finally the divergence between the two joint probability distributions is minimized using as a cost function the Kullback-Leibler divergence equation :

$$C = KL(P||Q) = \sum_i \sum_j p_{ij} log \frac{p_{ij}}{q_{ij}} \tag{2.21}$$

2.7 Metrics

In the idea of developing effective models within any field, be it machine learning, statistics, or computational modeling, it becomes imperative to assess the performance of these models

rigorously. This assessment not only validates the efficacy of the proposed models, but also provides information on their strengths and limitations. In the realm of computational biology, where the utilization of various models is prevalent, the evaluation of model performance is a critical step in order to ensure the reliability and applicability of the proposed solutions.
In this Section of the thesis, we dive into an exploration of the various metrics used for assessing the performance of used models. We examine the rationale behind the selection of these metrics and elucidate their mathematical formulations.

Precision :
Precision is a metric commonly employed in classification tasks; it quantifies the proportion of correctly identified positive instances among all instances predicted as positive by the model, thus exploit the capability of the model to discern relevant patterns from noise. Mathematically the formula is :

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives} \tag{2.22}$$

In the context of this thesis work this is an important metric since the False Positives can be unknown PET degrading proteins that are for example wrongly labeled as negative during the creation of the dataset used for training and test the models. That miss labeling is not due to negligence in the dataset creation but only from the lack of information during its creation, in fact it's not guarantee that the plasticDB contains all known PET degrading proteins although we assume that it contains most of them.

Recall :
Recall quantifies the proportion of correct identification of all positive instances within the dataset. Mathematically the formula is :

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} \tag{2.23}$$

This equation focus on maximizing the detection of true positive instances, irrespective of the presence of false negatives. A high recall score means a low rate of false negatives, indicative of the model's robustness in identifying all relevant instances within the dataset.

F1 score :
F1 score is a composite metric that harmonizes precision and recall into a single measure, emerging as a valuable tool for gauging the overall effectiveness of a model. The F1 score is a harmonic mean of precision and recall providing a holistic evaluation of a model ability to balance be-

tween minimizing false positives and false negatives. Mathematically the formula is :

$$F1\ score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.24}$$

This formula encapsulates the essence of the F1 score, which emphasizes both precision and recall. By harmonizing these two crucial aspects of model performance, the F1 score offers a balanced assessment that accounts for the interaction between false positives and false negatives.

Matthew's correlation coefficient :

The Matthew's correlation coefficient (MCC) takes into account true positives, true negatives, false positives, and false negatives to provide a balanced assessment of classification performance. Unlike metrics such as accuracy, which may be misleading in the presence of class imbalance, MCC considers the complete confusion matrix in order to yield a more comprehensive evaluation. The mathematical formula is :

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{2.25}$$

The MCC ranges from -1 to +1, where +1 indicates perfect classification, 0 indicates random classification, and -1 indicates total disagreement between prediction and observation. In our case, MCC provides valuable insights into the performance of the models, since the used data set is quite unbalanced.

2.8 Software

The entirety of the computational tasks in this thesis were performed using the Python programming language. Specifically, the subsequent libraries were utilized :

1. **Scikit-learn[29] :** used for the implementation of label propagation models and stratified 10-fold cross validation, computation of t-SNE, and performance metrics computation.
2. **NetworkX[30]** : used to obtain graphs from adjacency matrices.
3. **Karate Club[31] :** which contains different graph analysis tools. In particular, this was used to implement four graph embedding methods : Graph2vec, GL2vec, FeatherGraph and Wavelet (see Section **??**).
4. **Pandas [32]** and **Numpy [33]** : for data maninipulation.

3
Results

In this chapter we present the results of classifying PET-degrading proteins using three different approaches (Figure 3.1) :

1. **Sequence embedding approach:** through the use of ESM1b model convert the protein sequences into numerical vector. The idea in that approach is to use the amino acid sequence as a concatenation of letters in order to apply on it a natural language processing model (ESM1b).

2. **Graph embedding approach :** after converting each protein structure into a graph as we will see in the following section 3.2 through the use of a graph embedding model we convert each graph into a numerical vector. The idea behind that is try to extract structural information that is strictly associated to the function of a protein, assuming that proteins with similar function probably share also a similar function.

3. **Combined sequence and graph embedding approach**: As a final step, we merge the vectors from the sequence and structural approaches into a single vector. The idea behind that is to consider either sequence and structural informations in order to obtain a more comprehensive representation of the proteins.

As we saw in the data section2.1 our data set is composed of 9,883 proteins, 73 of which are involved in PET degradation. Our initial dataset has the characteristics of the small sample shown in Table 3.1, where in the first column we have the entry of the protein, in the second the amino acid sequence, and in the third one the EC number(see 2.1 for an explanation of that element).

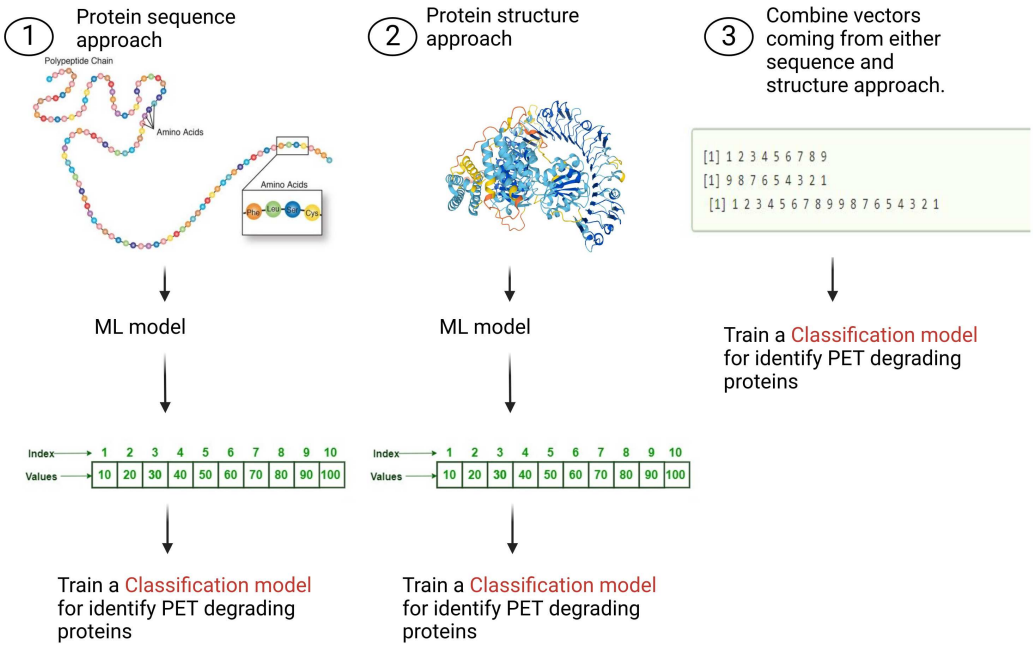

Figure 3.1: Different approaches used in order to obtain the embedding used to train classification models for the identification of PET-degrading proteins.

Entry	Sequence	EC number
Q9VP61	MPAEKSIYDPNPAI...	6.2.1.1
00073	MRGVWRYMPVYY...	3.1.1.74
A8G3E4	MVALRLIPCLDVA...	4.3.2.10
00083	MNFPRASRLMQA...	3.1.1.101
Q0AS12	MKEPAFWRTDGG...	2.7.1.130

Table 3.1: Starting dataset. obtained from PlasticDB and UniProt One thing to note is that the PET-degrading proteins have different entry nomenclature assignment with respect to the generic ones (entries with both letters and numbers) since they are extracted from two different databases.

In the upcoming sections, we will show the diverse classification outcomes achieved using the three distinct approaches visible in Figure 3.1, and also compare them.

3.1 Sequence embedding approach

From the data set of Table 3.1, the ESM1b model was applied as explained in the Methods Section 2.3.2. Since each amino acids can be one hot encoded by a single letter a protein sequence can be seen as a concatenation of letters. Therefore, the model converts amino acid sequences into numerical vectors (embeddings). For each protein, we thus obtain an embedding that is a one-dimensional numerical vector of 1,280 elements.

Initially, those vectors were used to produce a graphical idea of how sparse the data are and the distribution of the two classes in the embedding space. Due to the high number of dimensions in these embeddings, a dimensionality reduction technique was utilized to transform the data into a 2-dimensional form and create a graphical representation. Specifically, the t-distributed stochastic neighbor embedding (t-SNE - Section 2.6) was employed. The idea behind that was to assess whether PET-degrading proteins are localized in a specific region in the embedding space. If it is true, then we expect that a classification model will discriminate better the proteins of interest (the PET-degrading ones). In Figure 3.2, the colored points represent PET-degrading proteins, while the grey points represent generic proteins for which we do not focus on their function. Since in PET degradation there are different types of proteins with different functions, each color (which is based on EC number) represents a subset of PET degrading proteins (see Section 2.1 for an explanation of what an EC number is). Therefore, proteins with the same color in Figure 3.2 belong to the same function nomenclature specified by the EC number. Although PET-degrading proteins belong to different types of functions we notice that most of them tend to cluster in the upper right of the plot, suggesting that ESM1b assigns vectors to PET-degrading proteins that are near the embedding space and thus they likely tend to present some common sequence characteristics. However, several proteins depart from such a cluster, suggesting also that ESM1b identifies some latent sequence heterogeneity among PET-degrading proteins.

Classification model evaluation:

We aim to utilize label propagation for classification purposes. The rationale behind opting for a semi-supervised approach is that it is straightforward to acquire protein information such as sequence, while obtaining their function, which serves as the label in this case, is challenging and costly. Related to that, the final idea is to utilize unlabeled proteins obtained from heavily

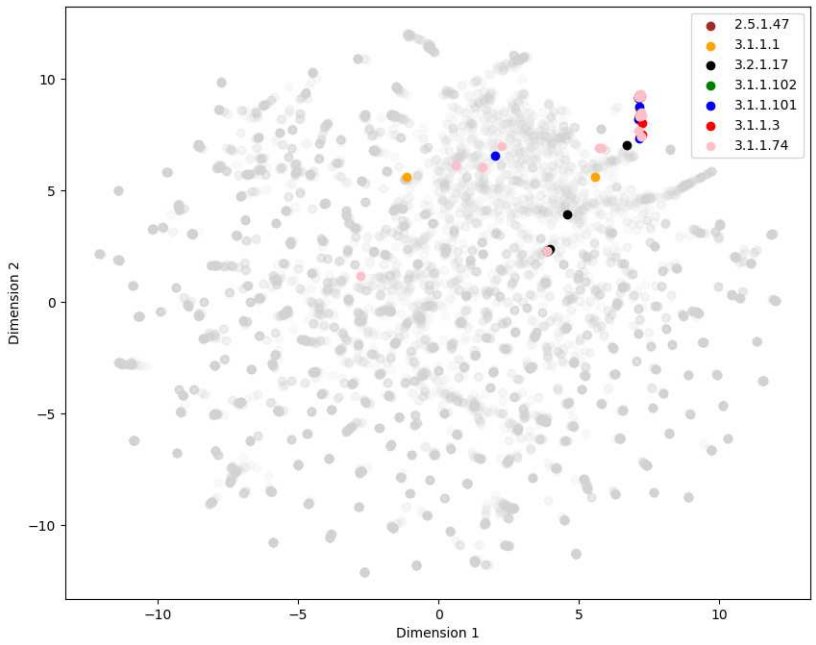

Figure 3.2: t-SNE representation of the embedding produced by ESM1b model. Proteins that are associated to PET degradation are colored based on their EC number, generic proteins are represented as grey points.

contaminated environments and incorporate their data for training the model, a task that is not feasible in a typical supervised approach.

Since we want to simulate a real-world situation to assess the model's performance on unlabeled data, we deliberately designated 10 random proteins as unlabeled (e.g., label=-1): 5 PET-degrading (true label=1) and 5 generic proteins (true label=0). Together, these samples represent the validation set used after the model hyper-parameter selection.

The concept is to assign a label of 1 to the vectors linked to PET degrading proteins, a label of 0 to those known to be unrelated to PET degradation, and a label of -1 to the remaining ones. Subsequently, a label propagation model will be trained to assign a 1/0 label to those labeled as -1. However, a challenge arises as the number of generic proteins exceeds that of PET-degrading proteins by a factor of 100. This significant class imbalance could lead to unreliable classification performance due to the unequal representation of the two classes. Another issue that may occur is that, within the general dataset, information on the proteins not linked to PET degradation is in general not available.

To solve these issues, we first undersampled generic proteins coming from the SwissProt database. In particular, for each PET degradation protein, we have extracted two random proteins. Second, in order to avoid as much as possible wrong-assigned labels, we excluded from the under-

sampled proteins those that share the first three digits of the EC numbers with the EC numbers set of PET degrading proteins. This means that, for the negative set, we considered only proteins with very distant biological functions with respect to PET-degrading proteins. In this way we, collected "likely negative" samples to form the negative set. Therefore, we considered all the SwissProt undersampled proteins as generic ones (i.e. label=0) so that it is possible to evaluate the models in a binary classification scenario. After those operations there were 214 proteins available, 73 PET degrading labeled as 1 and 141 generic ones labeled as 0.

Hyper-parameters tuning:
The amino acid sequences of the resulting dataset converted into vectors by the ESM1b model were firstly used to select the γ hyper-parameter of label propagation. We used scikit-learn[29] Label Propagation model with radial basis function (RBF) kernels. RBF is used to compute the edge weights between nodes in graph constructed by the model (for more details, refer to label propagation section 2.5). RBF is defined as :

$$K(x_i, x_j) = exp(-\gamma||x_i - x_j||^2), \tag{3.1}$$

where x_i, x_j are the data points in the embedding space, $||x_i - x_j||^2$ is the squared Euclidean distance and γ is a parameter that defines how much influence each training example has. In particular, γ is a hyper-parameter that determines the spread of the kernel : a small γ value produces a more restricted decision boundary while a large γ parameter produces a wider decision boundary. Therefore, the value of γ helps determine the similarity between the data points, which is crucial for propagating labels from labeled to unlabeled data during the training process. In particular we have taken into account three different gamma values : 5, 10, and 20.

To assess the optimal γ value, we performed a stratified 10-fold cross-validation for each parameter. We employ a stratified approach to ensure that each fold maintains an equal proportion of the two classes, enabling us to compare each fold with the others. There are some details that are important to point out about using label propagation in a 10-fold cross-validation scenario. In particular, at each fold all the labeled samples in the test set are masked as unlabeled (e.g. -1), then for the training of the model both the labeled and unlabeled data are used. Performance metrics are next calculated on the masked samples using their true and predicted labels. Furthermore, precaution is taken when one or more of the proteins initially masked as unlabeled are found in the test set of a fold. In such cases, we exclude their predicted labels from the computation of performance metrics, this is done to mimic a real-world scenario where

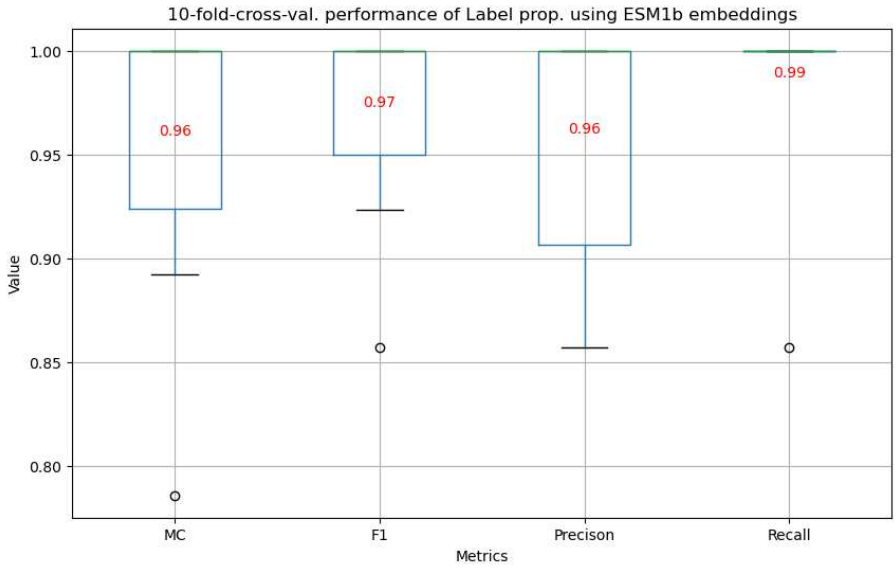

Figure 3.3: Boxplot of performance metrics resulting from a 10-fold-cross validation, using label propagation model with $\gamma = 10$ trained on vector resulting from ESM1b model. Values in red represent the mean of the metric in the corresponding box.

the true labels of these proteins are unknown. In Table 3.2 we can see the mean performance metrics values resulting from the 10-fold cross-validation over the studied γ values.

	$\gamma = 5$	$\gamma = 10$	$\gamma = 20$
mean MCC	0.90	0.96	0.91
mean F1 score	0.93	0.97	0.93
mean Recall	1	0.99	0.88
mean Precision	0.87	0.96	1

Table 3.2: Mean values of performance metrics resulting from a 10-fold-cross validation, using a label propagation model trained on vectors resulting from ESM1b model. Notes: MCC stands for Matthew's Correlation Coefficient.

As we can see from Table 3.2, the γ value that gives us the best performance based on the Matthew's Correlation Coefficient (MCC) is $\gamma = 10$. MCC is the most robust metric among those considered, as it takes into account both false positives and false negatives, as well as dataset imbalances, thus it was used to select the optimal hyper-parameter. After selecting the γ hyper-parameter, we further explored performance metrics by graphically visualizing the results of a 10-fold cross-validation using boxplots (Figure 3.3). The results in Figure 3.3 indicate that the model shows outstanding performance in identifying PET-degrading proteins, as indicated by the notably high MCC scores and F1 scores.

What we can observe from the box plots 3.3 is that the values of all metrics are skewed toward the upper part of the box plot. Therefore, the recall metric nearly reached perfection in each fold (with a mean value equal to 0.99). This suggests that the classification model produces very small number of false negatives. In this scenario, this is highly beneficial since we are sure that we do not miss out possible true positive e.g. PET degrading proteins, which can be advantageous for selecting and experimentally testing the potential degradation efficiency of a large number of proteins.

Model validation :

Finally, we validated sequence-based label propagation on the 10 proteins initially masked as unlabeled. This allows us to asses the robustness of the model towards hypothetical unseen data. In this phase, a new label propagation model was trained using the full labelling information of the other 204 proteins and the determined optimal γ. The performance on that set can be seen in the following table :

	MCC	**F1**	**Precision**	**Recall**
Performance metric	1.00	1.00	1.00	1.00

Table 3.3: Performances of the sequence-based model on the unlabeled test proteins.

As we can see, all the proteins are correctly predicted, confirming the excellent performance observed in hyper-parameter tuning (Figure 3.3). Although this last result might suffer from an over-estimation due to the small test set size, it seems to confirm the success of the hyper-parameter selection process.

3.2 Graph embedding approach

An alternative strategy to address this classification challenge involves adopting a structural perspective on proteins. This is particularly relevant to plastic depolimerization, as degrading enzymes require specific conformations to attach and break long and complex carbon chains. Specifically, implementing this approach involves constructing a graph representation for each protein and deriving a vectorized representation of the graph through the application of graph embedding models, as described in the graph-to-vector Section 2.4.

To construct a graph for each protein, we used PDB files containing the coordinates of all atoms of the protein structure. All PDB files used in this study were derived from AlphaFold predictions, sourced mainly from the AlphaFold[19] database or PlasticDB[18], as they contain predictions for almost all proteins considered. For the remaining proteins without structure we

used an available online Colab notebook of the AlphaFold model which allows us to predict missing structures starting from their amino acid sequence. In the construction of a protein graph representation (Figure 3.4), we decided to use amino acids as nodes, and physical contacts between them as edges. So, let $G = (V, E)$ be a graph representing a protein, where each node $v \in V$ is an amino acid (AA) and the interaction between amino acids is described by and edge $e \in E$. We consider the existence of a contact between two AAs if they respect at the same time two conditions :

1. **sequence separation:** two AAs are considered in contact if the sequence separation is greater than a specified threshold. This is based on the fact that we want to consider only contacts that are due to interaction and not sequential proximity. In fact proteins are 3D objects, thus even if AAs seem far apart when we look at their order in the protein's sequence, they can actually be close together in the protein's physical structure. Vice versa, if they are too close in the sequence, their interaction is prevented by physical space occupation.

2. **distance:** two AAs can be in contact if they are separated by an Euclidean distance that is less than a threshold distance defined in Angstrom (Å).

Each of these conditions defines a hyper-parameter for the graph embedding models. As we can see from Figure 3.4b, the general structure of an AA consists of three key components : an amine group, a carboxylic group, and an alpha-carbon.

1. **Amine group (NH_2):** the presence of this amine group is what gives amino acids their basic properties, as it can accept a proton (H^+) to become positively charged.

2. **Carboxylic group (COOH):** this group consists of a carbon atom double bound to an oxygen and single bound to a hydroxyl group (OH). The carboxylic group provides amino acids with acidic properties, since it can donate a proton (H^+) to become negatively charged.

3. **Alpha-carbon (α-carbon):** between the amine and carboxylic groups lies the $\alpha-$carbon. This central atom is bound to four different groups: the amine group, the carboxylic group, a hydrogen atom, and a variable "R" group. The "R" group, also known as the side chain, is what distinguishes one amino acid from another. It can vary greatly in size, shape, and chemical properties, determining the unique characteristics and functionality of each amino acid.

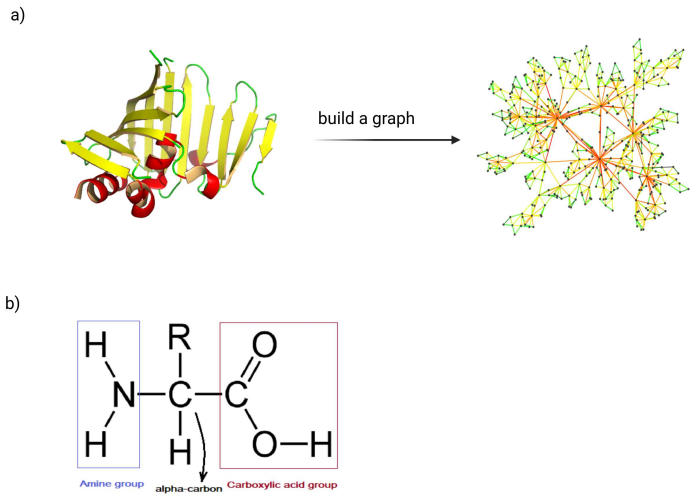

Figure 3.4: a) Transformation from protein to graph. b) General structure of an amino acid. The alpha carbon is centrally located and serves as a representative element in the structure for each amino acid.

Since the α-carbon atom is consistently present in all amino acids and has a central position, we can utilize its coordinates as a representative point for each amino acid and use it for computing the distance parameter. Once the interaction between all pair of amino acids that respect the given conditions is determined, a contact matrix is constructed. This matrix has a shape of $n \times n$ where n corresponds to the number of amino acids of the given protein. Each row and column corresponds to a specific residue in the protein sequence. The value at position (i, j) of the matrix is set to 1 if residues i and j are in contact and to 0 otherwise. The contact matrix so defined provides the adjacency matrix of a graph. In Figure 3.5 a heatmap graphical visualization of the adjacency matrix of a ribonuclease protein (SwissProt entry id:P23540).

Embedding model selection:

Among the existing graph embedding models, in particular we considered the following four models: WaveletCharachteristich, FeatherGraph, GL2Vec and Graph2Vec. All these methods were built in the KarateClub package[31]. Each of these models produces an alternative type of embedding vector for each protein, with potentially different effects on classification performance. Therefore, we initially tested these embedding models to select the best performing one. We thus built a graph for each protein from the same data set used in Section 3.1, containing 214 proteins, using each of the four tested embedding methods. The hyper-parameters used to built these graphs were kept fixed as: sequence separation$\geq$3 AAs and distance$\leq$6Å.

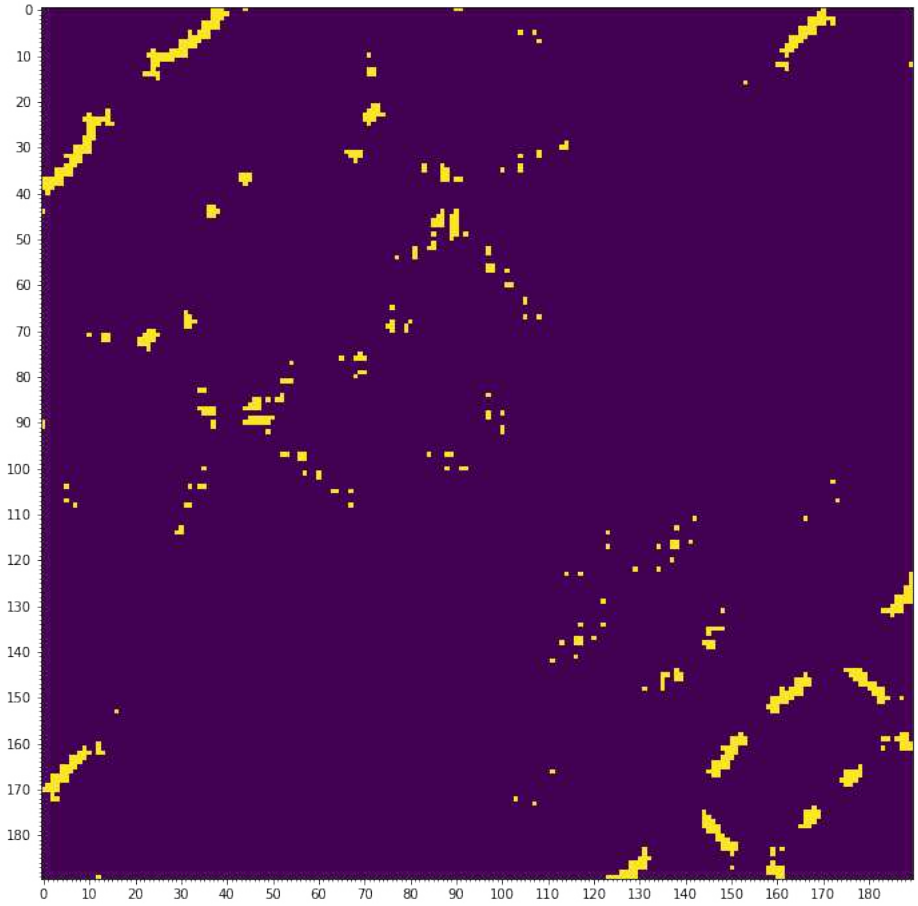

Figure 3.5: Adjacency matrix displayed as heatmap, where rows and columns represent amino acid residues and colored dot indicated the presence of contacts between residues.

With the same logic of the previous section (Section 3.1) we performed a label propagation evaluation through a 10-fold-cross validation, whose resulting averaged performance metrics can be seen in Table 3.4. As we can see from Table3.4, Wavelet, FeatherGraph and GL2Vec

	Wavelet	FeatherGraph	GL2Vec	Graph2vec
mean Mcc	0.75	0.77	0.73	0.39
mean F1	0.82	0.83	0.81	0.47
mean Acc	0.86	0.89	0.88	0.74

Table 3.4: Mean values of performance metrics resulting from a 10-fold-cross validation, using a label propagation model trained on vectors resulting from different graph embedding models. The parameters used for creating a graph are sequence separation ≥ 3 amino acids and distance ≤ 6 Angstrom

produce similar performances while Graph2Vec produces the worst performance. We selected the FeatherGraph model since in all considered metrics it produced the highest performances.

Hyper-parameters tuning:
Following the selection of the graph embedding model, we fine-tuned the hyper-parameters used to build the graphs and the γ value used by the label propagation model for the classification task. This was carried out with the aim of enhancing the classification performance of PET-degrading proteins. To focus on a reasonable number ofcombinations, we decided to use three different values for each parameter, in particular:

- **sequence separation** $\geq$ [2,3,4] AAs.
- **distance** $\leq$ [6,7,8] Å.
- γ **value**=[5,10,20]

In total 27 possible combinations of these three parameters were explored. For each combination of sequence separation and distance, we produced a graph embedding representation, the labeling of the graphs follows the logic seen in the previous Section 3.1. Ultimately, we also conduct the classification using the label propagation model by choosing the optimal value of γ. The 10-fold cross-validation follows the same rules presented in the previous section.
As before, the metrics considered in this scenario are : MCC, F1 score, precision and recall. After performing all 10-fold cross-validation, the parameters that achieved best performance results are sequence separation$\geq$2, distance$\leq$7 and $\gamma = 20$. By examining the boxplots in detail (see Figure 3.6), we can observe the varying classification performance using graph embeddings as data. In particular, we observe that overall performances are considerably lower

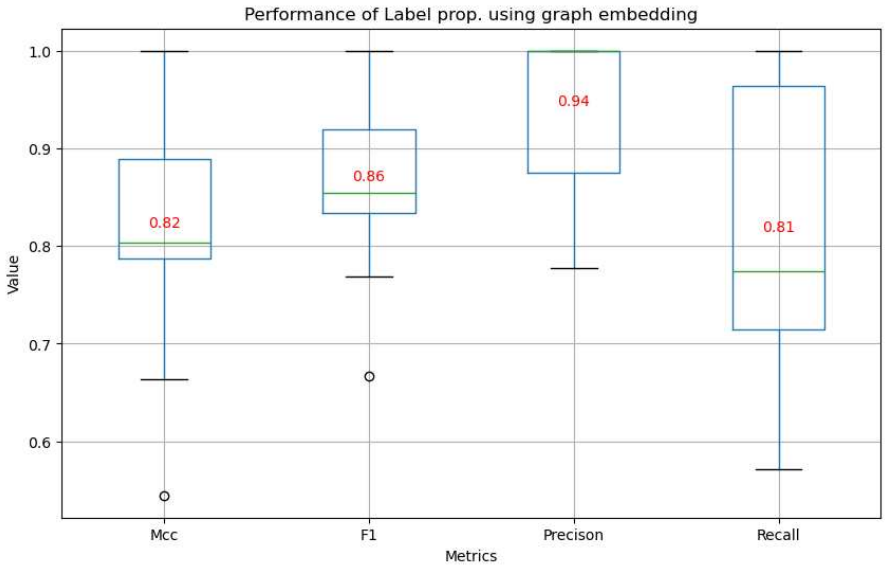

Figure 3.6: Boxplots of performance metrics resulting from a 10-fold-cross validation, using label propagation model with $\gamma = 20$ trained on vectors resulting from FeatherGraph model. The parameters used for building each graph are: sequence separation$\geq$2 and distance$\leq$7. Values in red represent the mean of the metric in the corresponding box.

compared to those achieved with vectors derived from the ESM1b model. Specifically, focusing on the recall boxplot, we observe a notable spread, indicating a highly variable number of false negatives generated at each fold of the cross-validation. Consequently, the model exhibits limited robustness in identifying PET degrading proteins with respect to previously obtained results (Section 3.3).

Model validation:

Like for the sequence-based model, we evaluated structure-based label propagation on the 10 proteins initially masked as unlabeled. The performances obtained on these unlabeled data can be seen in Table 3.3.

	MCC	**F1**	**Precision**	**Recall**
Performance metric	0.82	0.88	1.00	0.80

Table 3.5: Performances of the structure-based model on the unlabeled test proteins.

From Table3.5 we can see the performance on the initially set as unlabeled proteins, which are 5 generic proteins (label=0) and 5 PET-degrading proteins (label=1). Looking at the precision metric we can infer which all the generic proteins are correctly classified, while looking at the re-

call metric we can infer that only one PET degrading protein was misclassified. Therefore, the performances on that final test set are in line with those obtained in the hyper-parameter tuning stage and can be considered as satisfactory, despite being lower than those of the sequence-based model. The misclassified PET-degrading proteins is a cutinase with a PlasticDB entry of "00075". Cutinases are serine hydrolases that degrade cutin, a polyester of fatty acids that is the main component of plant cuticle. Interest in this specific group of enzymes has grown due to the discovery of some enzymes within this group that possess the ability to alter and break down PET.

3.3 Combined sequence and graph embedding approach

Finally, we integrated the sequence and structure of the proteins to leverage the information extracted from both approaches, to test whether it has complementarity beneficial to the classification performance. We combined the embedding data matrices generated by two models, ESM1b (Section 3.1) and FeatherGraph (Section 3.2), and then normalized them by column to preserve the scale of the original values. Specifically, we conducted a Min-Max normalization to scale the values in each column to a range of 0 to 1. Then with the same logic and values seen in the previous Section 3.6, we performed a grid search of the hyper-parameters used for constructing the graphs and the γ hyper-parameter to train label propagation. The hyper-parameters that achieve the best performamces are : sequence separation$\geq$4, distance$\leq$6 and γ=20. We can see in detail from the box plots in Figure 3.7 the performances obtained using those parameters. We can notice that the mean MCC parameter did not change with respect to the results obtained using only the ESM1b model (Section 3.1). In particular, we can notice that the recall boxplot is widely spread, suggesting that the combined approach introduces some noise, resulting in slightly worse performance than the sequence-only approach in the identification of PET degrading proteins. However, another aspect that we can notice is that the precision metric is always perfect at each fold of the 10-fold cross validation, and is superior to that of the other model types. This suggests that combining sequence and structural information generates a more stringent decision boundary, potentially useful when requiring a more stringent protein candidate set to test experimentally. Finally the performances of initially unlabeled set proteins are shown in Table 3.6. As we can see from the table, the combined approach correctly predicted all the initially masked protein, with the same performances as for the sequence embedding model 3.1. Also in this case, some over-estimation is possibly present, even though within the expected intervals based on the cross-validation results.

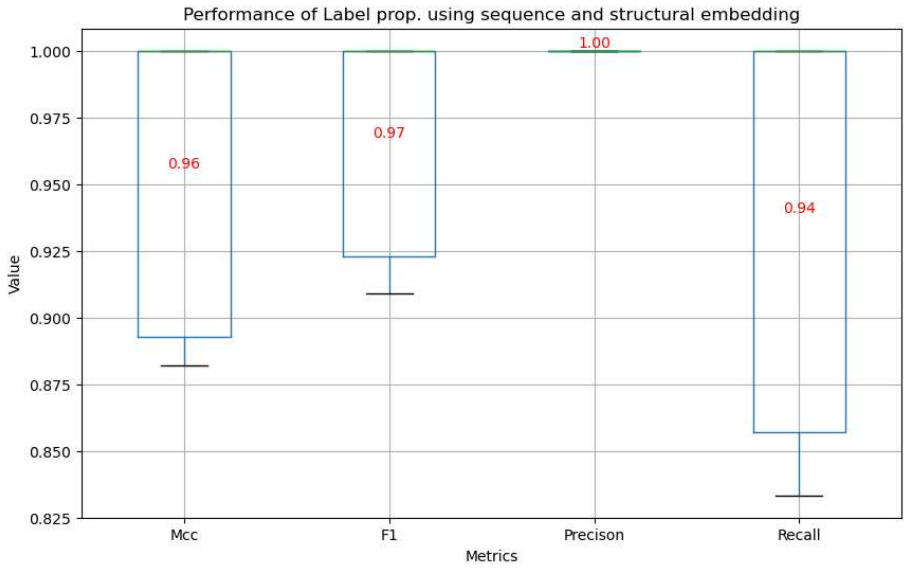

Figure 3.7: Boxplots of performance metrics resulting from a 10-fold-cross validation, using label propagation model with $\gamma = 20$ trained on a dataframe obtained from concatenation of ESM1b resulting vectors and FeatherGraph resulting vectors. The parameters used for create graphs are sequence separation$\geq$4 and distance$\leq$6. Values in red represent the mean of the metric in the corresponding box.

	MCC	F1	Precision	Recall
Performance metric	1.00	1.00	1.00	1.00

Table 3.6: Performances of the combined approach on the unlabeled test proteins.

4

Conclusion

In this work, we built and evaluated different machine learning models to predict the role of proteins in PET degradation, with potential application in discovering unidentified degradation enzymes. The results presented in Chapter 3 indicate that we have likely succeeded in meeting this goal : both the sequence and structural embedding approaches achieved good classification performance, in particular the sequence embedding approach ESM1b achieved excellent results. However, combining the two approaches did not improve these results, probably due to the minimal improvement margin and the limited validation set used.

Future development of the work can be to produce a larger training dataset containing additional PET-degrading proteins, some of which may not be validated, given the semi-supervised scenario we are currently operating in. Additionally it would be interesting to consider proteins that diverge in terms of structure and sequence with respect to the already known ones to consider a wider range of possible characteristics associated with PET degradation. In this way, a more robust assessment of the developed approach could be achieved.

Another line of investigation can be to consider a wider range of parameters for building the graphs and also explore more graph embedding approaches in order to potentially obtain also with that approach comparable performance of the one saw in the sequence approach (see Section 3.1).

Regarding the combined use of sequence and structural information, a different data integration approach could be used and tested. In particular, when both the sequence and structural approaches yield similar performance outcomes, instead of combining the resulting vectors

into a single one, an idea can be to interpolate the label propagation predictions of the two approaches and see if the misclassified proteins overlap. If, as suggested by our recall and precision results, the predictions have sufficient complementarity, then an ensmeble approach could be implemented and applied on unseen data in order to cover up a wider range of proteins that can be PET-degrading.
Finally, as a last step, trained classification models can be applied on unseen proteins extracted from landfills or environments with high PET contamination and validate the prediction through wet lab experiments, with the aim of discovering new PET-degrading proteins that, hopefully, due to the high concentration of PET evolve in order to efficiently degrade that material.

References

[1] J. e. a. Jumper, "Highly accurate protein structure prediction with alphafold." *Nature*, 2021.

[2] M. DS, C. LJ, and S. R. et al., "Protein 3d structure computed from evolutionary sequence variation." *PLoS One*, 2011.

[3] L. S. Wang and Qing, "A hybrid approach to recognize generic sections in scholarly documents," *International Journal on Document Analysis and Recognition (IJDAR)*, 2021.

[4] van Engelen and J. E., "A survey on semi-supervised learning," *Machine Learning*, 2020.

[5] R. Geyer, J. R. Jambeck, and K. L. Law, "Production, use, and fate of all plastics ever made," *Science Advances*, 2017.

[6] B. DK, G. F, T. RC, and B. M, "Accumulation and fragmentation of plastic debris in global environments," *Philosophical Transactions of the Royal Society*, 2009.

[7] D. Danso, J. Chow, and W. R. Streit, "Plastics: Environmental and biotechnological perspectives on microbial degradation," *Applied and Environmental Microbiology*, 2019.

[8] A. A. Shah, F. Hasan, A. Hameed, and S. Ahmed, "Biological degradation of plastics: A comprehensive review," *Biotechnology Advances*, 2008.

[9] J. Gu, "Microbial corrosion of metals and deterioration of polymeric materials," *Journal of Materials Engineering*, 1999.

[10] ASTM and AS, "Standard specification for labeling of plastics designed to be aerobically composted in municipal or industrial facilities," *non defined*, 2019.

[11] F. Awaja and D. Pavel, "Recycling of pet," *European Polymer Journal*, 2005.

[12] S. Yoshida, K. Hiraga, and T. T. et al, "A bacterium that degrades and assimilates poly(ethylene terephthalate)," *Science*, 2016.

[13] T. U. Consortium, "The universal protein resource (uniprot)," *Nucleic Acids Research*, 2007.

[14] Altschul, Gish, and M. et al., "Basic local alignment search tool," *J Mol Biol.*, 1990.

[15] Shroff, Raghav, Cole, and A. W. et al., "Discovery of novel gain-of-function mutations guided by structure-based deep learning," *ACS Synthetic Biology*, 2020.

[16] H. Lu, D. Diaz, and N. e. a. Czarnecki, "Machine learning-aided engineering of hydrolases for pet depolymerization," *Nature*, 2022.

[17] B. Fernández, G. Castillo, and Q. P. et al., "Microbial degradation of polyethylene terephthalate: a systematic review," *SN Appl. Sci.*, 2022.

[18] V. Gambarini, O. Pantos, and J. M. K. et al, "Phylogenetic distribution of plastic-degrading microorganisms," *mSystems*, 2021.

[19] V. Mihaly, A. Stephen, and D. et al., "Alphafold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models," *Nucleic Acids Research*, 2021.

[20] A. Vaswani, N. Shazeer, and N. P. et al., "Attention is all you need," *not specified*, 2023.

[21] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *not specified*, 2019.

[22] A. Rives, J. Meier, and T. S. et al, "Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences," *Proceedings of the National Academy of Sciences*, 2021.

[23] A. Narayanan, M. Chandramohan, and R. V. et al., "graph2vec: Learning distributed representations of graphs," *not specified*, 2017.

[24] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *not specified*, 2013.

[25] C. Hong and K. Hisashi", "Gl2vec: Graph embedding enriched by line graphs with edge features," *not specified*, 2019.

[26] B. Rozemberczki and R. Sarkar, "Characteristic functions on graphs: Birds of a feather, from statistical descriptors to parametric models," *not specified*, 2020.

[27] L. Wang, C. Huang, W. Ma, X. Cao, and S. Vosoughi, "Graph embedding via diffusion-wavelets-based node feature distribution characterization," *not specified*, 2021.

[28] Zhu and Xiaojin, "Semi-supervised learning literature survey," *Comput Sci, University of Wisconsin-Madison*, 2008.

[29] F. Pedregosa, G. Varoquaux, and A. e. a. Gramfort, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, 2011.

[30] H. Aric, P. Swart, and D. Chult, "Exploring network structure, dynamics, and function using networkx," *Proceedings of the 7th Python in Science Conference*, 2008.

[31] B. Rozemberczki, O. Kiss, and R. Sarkar, "Karate club: An api oriented open-source python framework for unsupervised learning on graphs," 2020.

[32] T. pandas development team, "pandas-dev/pandas: Pandas," *not specified*, 2020.

[33] T. numpy development team, "Array programming with numpy," *Nature*, 2020.

Acknowledgments

Thanks to Guido Zampieri for helping me in that work.