

國立臺灣大學工學院土木工程學系

碩士論文

Department of Civil Engineering

College of Engineering

National Taiwan University

Master Thesis



以蛋白質動力輔助圖神經網路預測蛋白質熱穩定性

Protein Thermostability Prediction by Graph Neural Network  
with Dynamics-Informed Graph Representation of Proteins

陳彥霖

Yen-Lin Chen

指導教授：張書瑋 博士

Advisor: Shu-Wei Chang, Ph.D.

中華民國 112 年 7 月

July 2023

# 致謝



踏進台大校園已然過了七年，取得碩士學位的我大概也進入了台大生活的最  
終章，剛好趁這個機會，說出一路走來未曾表達的感謝。不敢說自己多有長進，  
但我沒有成為自己討厭的樣子得感謝身旁的大家。

加入張書瑋教授的研究室也將近五年了，這段日子確實讓我對研究、對生活  
有些體悟。我要感謝指導老師張書瑋教授不厭其煩的指點、糾正我在研究上的問  
題和思想謬誤，在我研究卡關的時候提供關鍵的闖關道具，就像明智版的小叮噹。  
特別感謝羽白不嫌棄找我討論研究和各種話題，不吝於提供想法，並讓我見識研  
究室冷氣的實力。感謝維翰和我分享並討論研究結果。感謝江元和我聊研究和生  
涯規劃。感謝李登不厭其煩解釋 LAMMPS 腳本。感謝宏智早起和我吃早餐，確  
保我有起床。感謝研究室的其他成員，為我的生活增添不少色彩。另外，感謝  
ETH Zürich 的施智仁老師、高分所徐善慧老師、應力所周佳靚老師百忙中擔任口  
試委員，並在口試中提供洞見、想法和鼓勵。

我要感謝我的老爸、老媽和家人們，有你們的栽培和提供的資源，我才可能  
得到這樣的教育。感謝你們的教導和包容。生活上，我要感謝我的舅舅和舅媽在  
我遇到困境的時候提供不同的想法以及實質上的建議。感謝霽修用奔放的思想，  
讓我對生活和社會多了幾分領悟。感謝 Anton 願意讓我跟他們出去玩。感謝又中  
讓我理解文學美麗之處，並和我聊天、整理思緒。感謝易劭非常務實的討論。感  
謝承遠和恩代教我運動。感謝韋陵不忘找我吃飯（但約不到）。感謝宛純的陪伴。  
感謝冠寧和芳君的照顧。感謝所有朋友的包容。

碩士班最後幾個月裡，我大多時間都在研究室，甚至讓蘋果的「尋找」App  
以為土木研究大樓是我「家」。感謝拉拉不離不棄。

# 摘要



蛋白質之熱穩定性為蛋白質在極端溫度中維持可運作構型 (functional conformation) 的能力，並可被折疊態 (folded state) 與非折疊態 (unfolded state) 之間的雙態轉變 (two-state transition) 所量化。蛋白質於其原生環境溫度中大多呈折疊態，隨著系統升溫，非折疊態之蛋白質數量增加，此動態變動的過程中，當兩構型以相同數量存在的溫度即為蛋白質的熔點溫度 (melting temperature)，為蛋白質熱穩定性的重要指標。由於工業上時常需要將酵素置於非原生之高溫環境中，使得熔點溫度成為蛋白質工業適用性的重要指標之一，在設計或篩選工業酵素時為一大考量。

先前研究已證實將蛋白質之結構 (structure) 與動力 (dynamics) 特徵加譯為圖 (graphs) 並利用圖神經網路 (graph neural networks, GNN) 預測蛋白質功能的可行性。於此，本研究串聯蛋白質熱穩定性、蛋白質功能性、蛋白質結構動力資訊，展示如何將蛋白質的結構與動態資訊用於蛋白熔點溫度之預測。為了泛用於尚未解出實驗結構的蛋白質，本研究採用 AlphaFold 的預測結果作為蛋白質的結構，再以此結構為基礎，建立扭矩網路模型 (torsional network model, TNM)，並根據此力學模型獲得其簡正模態 (normal mode)，提供後續蛋白質動力耦合 (dynamic coupling) 計算。最終，蛋白質結構將以接觸圖 (contact graph) 和 PAE 圖 (predicted aligned error graph) 表示，蛋白質動態資訊則以共向圖 (co-directionality graph)、協調圖 (coordination graph)，以及變位圖 (deformation graph) 表示。結果顯示，將蛋白質經過以上處理加譯為圖，搭配圖神經網路預測熔點溫度，與實驗量測結果比較，平均絕對誤差為  $3.291^{\circ}\text{C}$ ，方均根差為  $4.286^{\circ}\text{C}$ ，而  $R^2$  可達 0.805。本研究亦利用影像辨識中特徵視覺化的技術，可反向檢視蛋白質中哪些殘基 (residues) 對於模型的預測有較高的影響力，亦可辨別資料中各種圖之於模型預測的重要程度。這些資訊再次指出蛋白動態對於熱穩定性的重要性，

並提供了改善熱穩定性的可能關鍵區域，可作為提升蛋白質工業應用表現之參考，亦為未來研究提供指引。




**關鍵字：**蛋白質、熱穩定性、熔點溫度、簡正模態、圖神經網路、深度學習、  
回歸激發圖

# Abstract



Protein thermostability, the resistance or preservation of protein functions under extreme temperatures, plays a vital role in numerous biotechnological applications. Since designed proteins, such as industrial enzymes and biocatalysts, are often subjected to temperatures that significantly differ from the cellular environment, protein thermostability has always been critical to consider when making protein designs or searching for proteins suitable for a specific task. Commonly simplified as a two-state transition, protein thermostability is primarily characterized by the melting temperature, where the folded and unfolded states are equally favorable. This work focuses on the prediction of the melting temperature of protein. As protein dynamics is essential in understanding protein functions, an effective data representation that includes the dynamics should benefit the melting temperature prediction. In this work, a graph-based (as in graph theory) representation of proteins that encompasses the protein sequence, structure, and dynamics is presented. A graph neural network architecture that uses message passing layers was designed to accommodate multiple types of connections. Protein structures were computed by AlphaFold, and the dynamics were computed based on the torsional network model (TNM)



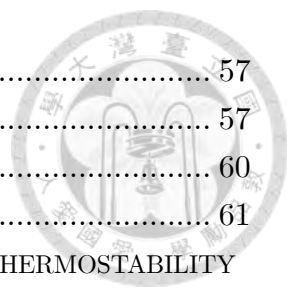
for training. Hence, the learned features and parameters can be readily applied to protein sequences without known experimental structure, satisfying the goal of aiding the prediction of design proteins. Critical regions that strongly influence the thermostability of proteins are identified by computing a graph regression activation map (RAM), which is based on the partial derivative of the predicted value with respect to the convolutional features map. The method provides an efficient approach to accessing the thermostability of new protein sequences. Further, it provides insights into the inner workings of proteins by identifying residues critical to thermostability.

**Keywords:** proteins, thermostability, melting temperature, normal mode analysis (NMA), graph neural networks (GNN), deep learning, regression activation map (RAM)

# Table of Contents



致謝 .....	I
摘要 .....	II
ABSTRACT.....	IV
TABLE OF CONTENTS.....	VI
LIST OF FIGURES .....	VIII
LIST OF TABLES .....	X
<b>CHAPTER 1. INTRODUCTION AND BACKGROUND.....</b>	<b>1</b>
1.1 PROTEIN THERMOSTABILITY .....	1
1.1.1 Definitions .....	2
1.1.2 Relation with Protein Dynamics and Function .....	4
1.1.3 Protein Thermostability Prediction .....	5
1.2 MACHINE LEARNING AND PROTEINS.....	8
1.2.1 ProteinBERT .....	8
1.2.2 AlphaFold .....	9
1.2.3 Graph Neural Networks in Protein Sciences.....	10
<b>CHAPTER 2. METHODOLOGY .....</b>	<b>13</b>
2.1 TORSIONAL NETWORK MODEL AND DYNAMICAL COUPLING GRAPHS .....	14
2.2 MAPPING PROTEINS TO GRAPHS.....	21
2.2.1 Graph Theory .....	21
2.2.2 The Data Representation.....	23
2.3 DATA SOURCE AND FEATURE DISTRIBUTION .....	30
2.3.1 Data Processing.....	30
2.3.2 Feature Analysis .....	33
2.4 THE NEURAL NETWORK MODEL .....	40
2.4.1 Graph Neural Networks.....	40
2.4.2 Architecture.....	43
2.4.3 Training Setup .....	46
2.5 REGRESSION ACTIVATION MAP (RAM).....	47
<b>CHAPTER 3. RESULTS AND DISCUSSION.....</b>	<b>49</b>
3.1 THE GRAPH REPRESENTATION OF PROTEINS .....	49



3.2 MODEL PERFORMANCE .....	57
Graph Representation is Effective in $T_m$ Prediction .....	57
Performance is Similar to the State-of-the-Art Method .....	60
Bias and Variance of the Model .....	61
3.3 STRUCTURAL AND DYNAMICAL INFLUENCE ON PROTEIN THERMOSTABILITY .....	65
<b>CHAPTER 4. CONCLUSION AND OUTLOOK .....</b>	<b>73</b>
FUTURE WORK .....	74
<b>BIBLIOGRAPHY .....</b>	<b>76</b>

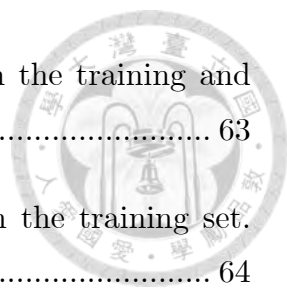




# List of Figures

<b>Figure 1.</b> The protein stability curve of a hypothetical protein. ( <i>Figure adapted from Pucci et al. [8]</i> ).....	3
<b>Figure 2.</b> Torsion angles on the protein backbone. ( <i>Figure adapted from the online course “The Principles of Protein Structure” organized by Birkbeck College [52]</i> ).....	17
<b>Figure 3.</b> Dataflow for building the graph representation of proteins. ....	28
<b>Figure 4.</b> Showcase of the six types of graphs defined in this work.....	29
<b>Figure 5.</b> A significant correlation exists between the optimal growth temperature $T_{og}$ and melting temperature $T_m$ . ....	32
<b>Figure 6.</b> The distribution of melting temperature $T_m$ and sequence length. ....	34
<b>Figure 7.</b> Correlation is between sequence length and melting temperature $T_m$ .34	
<b>Figure 8.</b> Distribution of pLDDT and B-factor of all residues from all proteins. ....	36
<b>Figure 9.</b> Correlation between B-factor/pLDDT and the melting temperature $T_m$ . ....	37
<b>Figure 10.</b> Correlation between sequence length and pLDDT/B-factors.....	39
<b>Figure 11.</b> The neural network architecture of the DYN model trained. ....	46
<b>Figure 12.</b> Distribution of the number of edges for all edge types. ....	52
<b>Figure 13.</b> Relationship between the number of edges in each protein and $T_m$ . 54	
<b>Figure 14.</b> Correlation between sequence length and the number of edges ..... 56	
<b>Figure 15.</b> Learning curve ( $R^2$ ) of the four models trained in this work. ....	59

<b>Figure 16.</b> True $T_m$ versus predicted $T_m$ of the OGT model on the training and validation dataset. ....	63
<b>Figure 17.</b> True $T_m$ versus predicted $T_m$ of the DYN model on the training set. ....	64
<b>Figure 18.</b> True $T_m$ versus predicted $T_m$ of the DYN model on the validation set. ....	65
<b>Figure 19.</b> RAM for the six types of graphs for a protein. ....	67
<b>Figure 20.</b> Contribution of the six different graphs to the prediction of $T_m$ .....	69
<b>Figure 21.</b> Contribution of each type of graph to the prediction versus $T_m$ . ....	71



# List of Tables



<b>Table 1.</b> Summary of the features for the graph representation of proteins. ....	28
<b>Table 2.</b> The 12 species included in the dataset, their optimal growth temperature $T_{og}$ , and the number of proteins. ....	31
<b>Table 3.</b> Overview of different models trained. ....	44
<b>Table 4.</b> Statistics on the various edge types. ....	50
<b>Table 5.</b> Performance of the four models. ....	58
<b>Table 6.</b> Comparison of performance between DYN and DeepSTABp. ....	61

# Chapter 1.

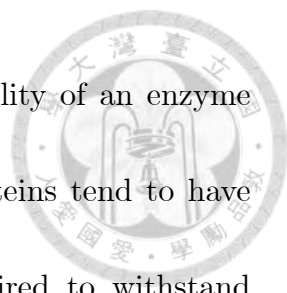


## Introduction and Background

Concepts and technology that this study builds upon are reviewed in this chapter. Section 1.1 introduces the basic concepts of protein thermostability and its link with protein dynamics. A review of recent advances in the prediction of protein thermostability is given at the end of the section. Section 1.2 highlights the success of integrating machine learning methods with protein science. Finally, how the ideas introduced in this chapter can be brought together is surfaced at the end of the chapter.

### 1.1 Protein Thermostability

Protein thermostability is the resistance of a protein to function loss under extreme temperatures. It is an important factor to consider in biocatalysts, where enzymes hold an advantage over traditional industrial catalysts due to their superior selectivity [1], albeit being required to withstand the non-physiological environments that are oftentimes necessary for the application. Not only does high thermostability of a protein ensures workability in these environments, it also translates to reusability of the biocatalyst and enables faster reaction rates [2].



These implications made thermostability a measure of suitability of an enzyme for industrial settings [3]. Furthermore, naturally evolved proteins tend to have poor thermostability without the evolutionary pressure required to withstand extreme heat [4, 5], adding to the importance of identifying high thermostability in the search for industrial enzymes [2].

The remainder of this section covers the definition and importance of the melting temperature  $T_m$ , the importance of protein dynamics to  $T_m$ , and, lastly, recent works on the prediction of melting temperature.

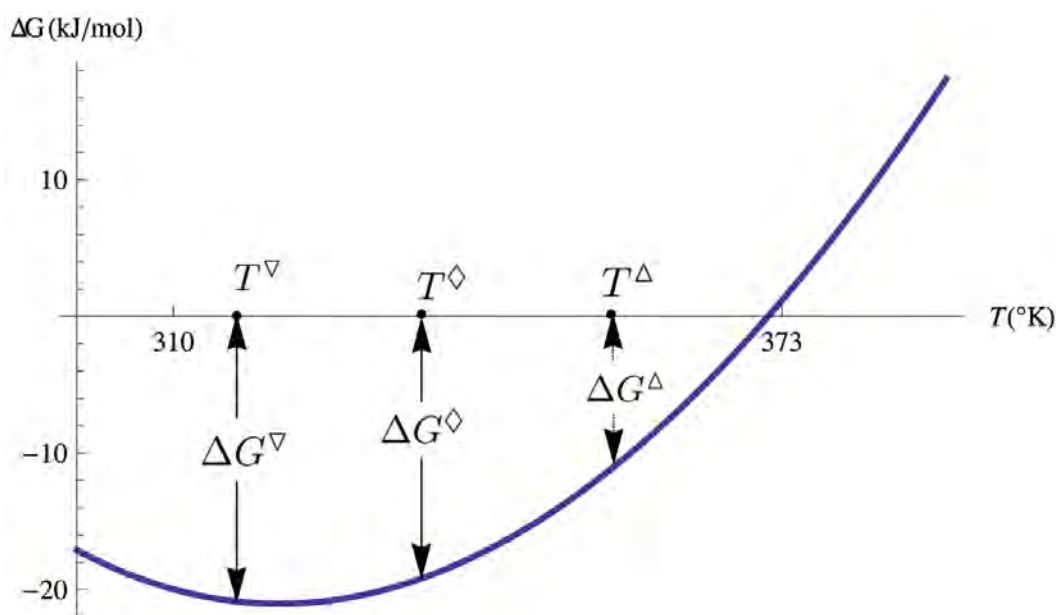
### 1.1.1 Definitions

The thermodynamic stability of proteins under extreme temperatures is commonly approximated as a reversible two-stage process, where the biomolecule transits from a functional "folded" conformation to a dysfunctional "unfolded" conformation [6]. Under native temperature, the folded state is energetically favorable when compared to the unfolded state. As the temperature deviates to a certain extent, the unfolded state would gradually become favorable and the protein loses its functional conformation.

Thermodynamically, the favorability of the two states can be succinctly described by Gibbs free energy  $G$ . Under constant pressure, the Gibbs free energy



is a function of temperature and can theoretically be determined for both the folded state ( $G_f$ ) and unfolded states ( $G_u$ ). The difference between the two states is known as the standard folding free energy  $\Delta G = G_f - G_u$  and is characterized by the Gibbs–Helmholtz equation [7]. Since the only variable here is the temperature  $T$ ,  $\Delta G$  can be plotted against  $T$  as the protein stability curve, as illustrated in Figure 1. The state with a smaller value of  $G$  is favorable to the system, i.e., a negative  $\Delta G$  is reflective of a temperature where the folded state is preferable, and vice versa. It should be noted that thermodynamical stability refers to the energetic difference between the two stable states, therefore, saying nothing about the height of the energy barrier (kinetic stability) or the likelihood of the transition between the two states.



**Figure 1.** The protein stability curve of a hypothetical protein. The melting

temperature  $T_m$  is the temperature where  $\Delta G = 0$ , or the temperature at which the folded and unfolded conformation are thermodynamically equally likely.

(Figure adapted from Pucci et al. [8])

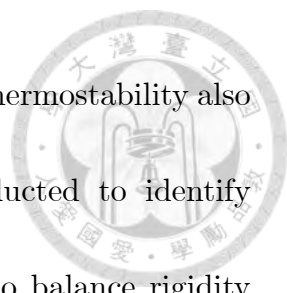


## Melting Temperature $T_m$

The melting temperature is defined as the temperature where the folded state and the unfolded state are in equilibrium, i.e., the temperature where half of the proteins are in the unfolded state. Graphically, this is where the stability curve crosses the x-axis in Figure 1. Going from the native temperature toward high temperatures,  $T_m$  corresponds to the temperatures at which the unfolded state becomes as favorable as the folded state, and heating beyond this point results in a system where the unfolded state is preferable. As such, the melting temperature is a characteristic index for protein thermostability and is commonly targeted for thermostability prediction.

### 1.1.2 Relation with Protein Dynamics and Function

Thermostability is associated to the functionality of proteins (loss of functionality to be precise), which is in turn heavily related to protein dynamics. Numerous works have explored the association between protein dynamics and protein functions. For instance, dynamics has been linked with allostery [9-11] and



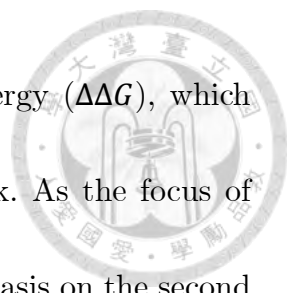
ligand binding [12-14]. The correlation between dynamics and thermostability also received attention. For example, MD simulations were conducted to identify regions of low thermostability within a protein [15]. A need to balance rigidity and flexibility was suggested to build thermostable proteins [16, 17].

However, it is difficult to make high-accuracy observations of the dynamics of proteins by experiments. Simulations that are able to sample the conformation space, such as molecular dynamics (MD) or Monte Carlo (MC) simulations have proved to be very useful, especially as computational technology advances [18]. Alternatively, normal mode analysis can be done on a surrogate mechanical model of the protein, known as elastic network models (ENM), to reduce computational costs. These models typically simplify each residue of the protein as a single mass point and connect these mass points with elastic springs. Due to its greatly superior computational efficiency, ENM analysis is employed in this work to extract dynamical information of proteins.

### 1.1.3 Protein Thermostability Prediction

Protein stability prediction can be categorized into two main categories: 1) the change in stability when the protein undergoes mutation and 2) the intrinsic stability of a protein. In the first task, common targets of prediction are the change





in melting temperature ( $\Delta T_m$ ) or change in unfolding free energy ( $\Delta\Delta G$ ), which corresponds to the prediction of  $T_m$  and  $\Delta G$  in the second task. As the focus of this study is on the prediction of  $T_m$ , this section will put emphasis on the second task.

Earlier attempts in predicting the intrinsic thermostability of proteins suffers greatly from the limited size of available databases, and mostly employs statistical methods on a wide array of protein-level features that are computed for each protein. The frequency of residues and amino acid composition (AAC) of the protein sequence is among the first features to be included for predicting  $T_m$  [19, 20]. Pucci et al. published multiple works that rely on linear regression of  $T_m$  based on statistical potentials and other factors such as growth temperature, B-factors, and sequence similarity [8, 21]. They later published a study that predicts the entire stability curve based on protein structure, the type of host organism, and the factors defined in their previous work [7]. This is the only attempt in generating the entire stability curve to date. It was found among these attempts, and also reported in at least one standalone study [22], that the optimal growth temperature is highly correlated to  $T_m$ . More recently, Miotto et al. proposed a parameter-free perturbation method that operates on a graph with a similar construct to the protein contact map in 2018 [23].

# Machine Learning Methods



Only a handful of methods makes use of the advances in data mining to take advantage of the recent large thermostability databases, such as the meltome atlas [24] or ProThermDB [25]. Yang et al. published ProTstab in 2019, a regression-tree based method that makes use of the 2,077 features that covers a wide range of physicochemical, structural, and compositional properties of each protein [26]. This was followed by ProTstab2 also from Yang et al., utilizing computational power to mine the correlation between  $T_m$  and 6,935 different derived features of over 30k proteins. A wide range of algorithms was compared in the work, including decision tree, random forest, support vector regression, gradient boost regression tree, extreme gradient boosting, light gradient boosting machine, and multi-layer perceptron regressor [27]. Most recently, DeepSTABp was published in April of 2023, where a neural network model was designed to predict  $T_m$  based on the ProteinBERT sequence embedding [28]. DeepSTABp is the only study that is based on deep learning, and its root mean squared error (RMSE) leads the other methods by a margin of at least 2°C.

None of these machine learning methods for  $T_m$  prediction utilized dynamical information as its features despite it has been show that protein dynamics are

closely linked to its functionality and thermostability. The aim of this study is therefore to incorporate protein dynamics into the machine learning methods and to leverage the additional information.



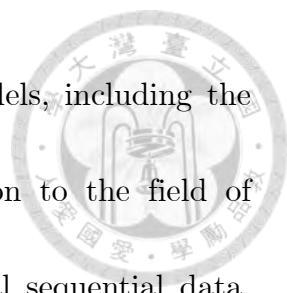
## 1.2 Machine Learning and Proteins

Machine learning (ML) is a branch of artificial intelligence (AI) that focuses on developing algorithms that can make inferences based on patterns learned from data without explicit programming. It has emerged as a powerful field of study in recent decades, revolutionizing various domains with its outstanding ability to extract hidden patterns in convoluted data structures that are oftentimes too entangled to be detected by human intelligence.

This section introduces some key ML models in the field of protein bioinformatics. A model to process protein sequences and a model that predicts protein structure are introduced. This is followed by a review of applications of graph neural networks on protein data, and finally the core idea and practical value of transfer learning.

### 1.2.1 ProteinBERT

The boom of natural language models (NLP) in language-related tasks

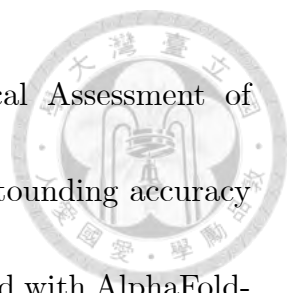


ushered the development of exciting sequence processing models, including the famed Transformers that introduced the concept of attention to the field of machine learning [29]. Fusion of these method with biological sequential data, such as protein or DNA sequences, followed quickly. Some of these breakthrough models includes TAPE-Transformer [30], ProtTrans [31], UniRep [32], and ProteinBERT [33].

ProteinBERT is a transformer-based neural network pre-trained on amino acid sequences and Gene Ontology (GO) functions. During the training process, the encoder of the model is forced to learn an embedding of the protein sequence that enables the decoder part to reconstruct the original sequence and the Gene Ontology (GO) annotations. The model, published in 2022, gained some traction as a sequence embedder/feature extractor in protein-related tasks, such as toxicity prediction [34], DNA-binding prediction [35], transporter protein identification [36].

## 1.2.2 AlphaFold

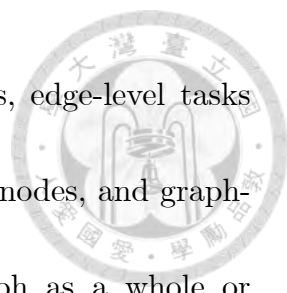
Protein structure prediction, the task of mapping protein sequences to their three-dimensional structures, has been a longstanding problem in biochemistry. One of the most significant breakthroughs for this task is the development of



AlphaFold from Google's DeepMind, whose debut in Critical Assessment of Structure Prediction (CASP) of 2018 made headlines for its astounding accuracy [37, 38]. In partnership with EMBL-EBI, a database was released with AlphaFold-predicted structures of over 300 million entries covering most of the protein sequences on UniProt [39]. The database provided a convenient source of protein structure that eliminated the need for end users to run the AlphaFold algorithm. More importantly, it mitigates the need for experimental structure by providing structures that can be acceptably accurate for downstream tasks.

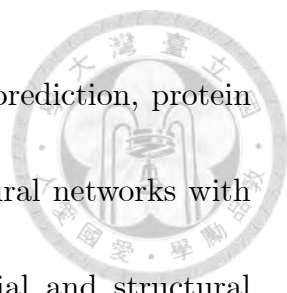
### 1.2.3 Graph Neural Networks in Protein Sciences

Graphs are natural descriptors of non-Euclidian data structures useful in modeling the relationships (edges) between a set of entities (vertices), not necessarily of the same type, commonly found in real-world data. Neural networks designed to process graph structure data are termed graph neural networks (GNN). Most of these architectures employ a message passing scheme [40], where the node features are treated as messages and edges as the passage along which the messages are propagated. Node messages are iteratively updated as a function of incoming messages from its paths. Typical tasks for graph neural networks can be classified into three main categories: node-level, edge-level, and graph-level [40,



41]. Node-level tasks typically predict the properties of nodes, edge-level tasks predict the existence or the label of edges between any pair of nodes, and graph-level tasks make inferences about the properties of the graph as a whole or generate new graph structures.

Numerous works have shown that graphs are effective representations of proteins, where vertices commonly model atoms or residues, and edges model interaction between the nodes, such as covalent bonds, hydrogen bonds, or disulfide bonds. Physical simulations, such as MD, can be accelerated by graph-based networks [42]. Protein design operates on graph structures for its effective description of the topology [43]. Graphs were also utilized to evaluate the performance of protein docking methods [44], predict protein-ligand binding affinity [45], or identify binding residues [46]. There were also attempts to determine whether two proteins interact based on their graph representation [47]. Proteins as graphs were also successfully adapted to the task of protein function prediction [48]. Dynamical information encoded as graphs is shown in the work of Yuan et al. to aid the performance of a GNN in protein function prediction [49].



In light of the recent breakthroughs in protein structure prediction, protein sequence embedding, and the successful marriage of graph neural networks with the protein molecule, this work aims to incorporate sequential and structural information via graph neural networks to predict the melting temperature of individual proteins. Also, considering the importance of dynamics to protein functionality, another objective of this work is to incorporate protein dynamics as a feature for the prediction. Finally, the regression activation map is computed to improve the interpretability of the neural network. The input and the architecture are described in detail in the next chapter.

# Chapter 2.

## Methodology



As outlined in the previous chapter, the properties of proteins are embedded in their sequence, structure, and dynamics. It has been sufficiently shown in the literature that not only do graphs serve well in presenting protein structures and dynamics, but they can also be efficiently and effectively processed by specific machine learning models, namely GNN. Therefore, predicting protein properties by GNN based on graph-structured data is a promising approach. This work leverages this combination for melting temperature prediction.

This chapter covers the workflow of this study. Section 2.1 describes how dynamical information is extracted from protein structures. Section 2.2 lays out the definition of the graph representation of proteins. Section 2.3 describes the architecture of the graph neural network used in this work. Lastly, section 2.4 gives the training configuration and hyperparameters for training the model.

Code to reproduce this work is available as a repository on GitHub:

<https://github.com/yenlin-chen/ai-thermostability>

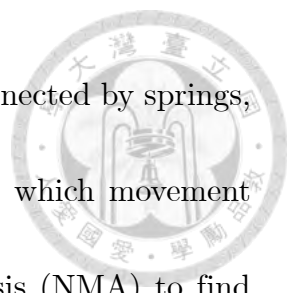


## 2.1 Torsional Network Model and Dynamical Coupling Graphs



As stated in the introduction, protein dynamics is a key aspect of protein science. Uncovering the movements of proteins and their interactions with surrounding molecules is a critical aspect in understanding their behavior. Experimental methods suffer from limited resolution due to the small size of these biomolecules, hence the study of protein dynamics relies heavily on computational methods, and a mechanical model for the system of interest must be built. Different models were proposed by making different assumptions or simplifications, and different analyses are conducted based on the specific advantages and drawbacks of each model.

Elastic network models (ENM) are a family of surrogate models commonly employed to characterize the motion of proteins. These models simplify the molecule as a system of mass points connected by springs. The potential surface computed from this model is a function of the conformation of these mass points. Degree of freedoms (DoF) of the system are also specified by the model, which is equivalent to constraints on the available movements. In summary, different ENM models might adopt different strategies to determine 1) how the protein structure



is mapped to a set of mass points, 2) which mass points are connected by springs, 3) the stiffness of each of the springs, and 4) the DoFs along which movement is parametrized. This is usually followed by normal mode analysis (NMA) to find an orthogonal basis of oscillation that spans all possible motions of the mass-and-spring model, analogous to Fourier analysis in signal processing. The following will describe the potential model of torsional network model (TNM) and the corresponding NMA procedure [50].

Given an all-atomic structure of a protein with  $n$  amino acids in equilibrium, denote the mass of the  $i$ -th atom by  $m_i$  and the three-dimensional Cartesian coordinates by  $\vec{r}_i = (q_{3i-2}, q_{3i-1}, q_{3i})$ . Denote the equilibrium distance of the closest pair of atoms between residue  $k$ -th and  $l$ -th by  $r_{kl}^0$  and the instantaneous distance between the two atoms by  $r_{kl}$ . Also denote the position of equilibrium of the  $i$ -th atom by  $\vec{r}_i^0$ . TNM assigns a spring to every pair of residues whose  $r_{kl}^0$  is smaller than a predefined cutoff  $c_{\text{TNM}}$ , which is set to 4.5 Angstroms in this work.

The spring constants are inversely proportional to  $r_{kl}^0$  by a power of 6

$$\gamma_{kl} \propto (r_{kl}^0)^{-6} \quad (2.1)$$

which the authors of TNM claim to yield optimal results. Hence, the potential of the spring-and-mass system (setting the equilibrium potential to zero) is

$$U = \frac{1}{2} \sum_{kl} R_{kl} \gamma_{kl} (r_{kl} - r_{kl}^0)^2 \quad (2.2)$$



where  $R_{ij}$  equals 1 if  $r_{ij}^0 \leq c_{\text{TNM}}$  and is 0 otherwise. The expression is summed over all pairs of residues.

DoFs of TNM are the torsion angles  $\Phi$  and  $\Psi$  of the backbone Figure 2, hence the total number of DoFs is  $2n$ . Denote the torsion angles as  $\theta_a$ , where  $a \in \{1, 2, \dots, 2n\}$ , and the axis of rotation of  $\theta_a$  as the unit vector  $\vec{e}_a$  (N-C $_{\alpha}$  bond for  $\Phi$  angles; C $_{\alpha}$ -C bond for  $\Psi$  angles). A change in  $\theta_a$  will affect the set of atoms  $\Omega_a$  upstream of  $\vec{e}_a$ . The derivative of  $\vec{r}_i$  with respect to  $\theta_a$  is therefore  $\partial \vec{r}_i / \partial \theta_a = \xi_{ia} \vec{e}_a \times (\vec{r}_i - \vec{s}_a)$ , where  $\xi_{ia}$  is 1 if the  $i$ -th atom belongs to  $\Omega_a$  and 0 otherwise, and  $\vec{s}_a$  is the origin of rotation (position of N for  $\Phi$  angles; position of C $_{\alpha}$  for  $\Psi$  angles). To ensure that the center of mass and the angular momentum of the protein is kept unaltered, the Eckart conditions are imposed on the Jacobian [51]

$$\sum_i m_i \vec{J}_{ia} = \vec{0} \quad (2.3)$$

$$\sum_i m_i \vec{r}_i \times \vec{J}_{ia} = \vec{0} \quad (2.4)$$

where  $\vec{J}_{ia}$  is the shorthand notation for the vector

$$\frac{\partial \vec{r}'_i}{\partial \theta_a} = \left( \frac{\partial q'_{3i-2}}{\partial \theta_a}, \frac{\partial q'_{3i-1}}{\partial \theta_a}, \frac{\partial q'_{3i}}{\partial \theta_a} \right) \quad (2.5)$$

with  $\vec{r}'_i = (q'_{3i-2}, q'_{3i-1}, q'_{3i})$  denoting the Cartesian coordinates of the  $i$ -th atom after Eckart's mass and angular momentum correction. The conditions induce translation  $\vec{\tau}_a$  and rotation  $\vec{\mu}_a$  for  $\theta_a$ . This results in

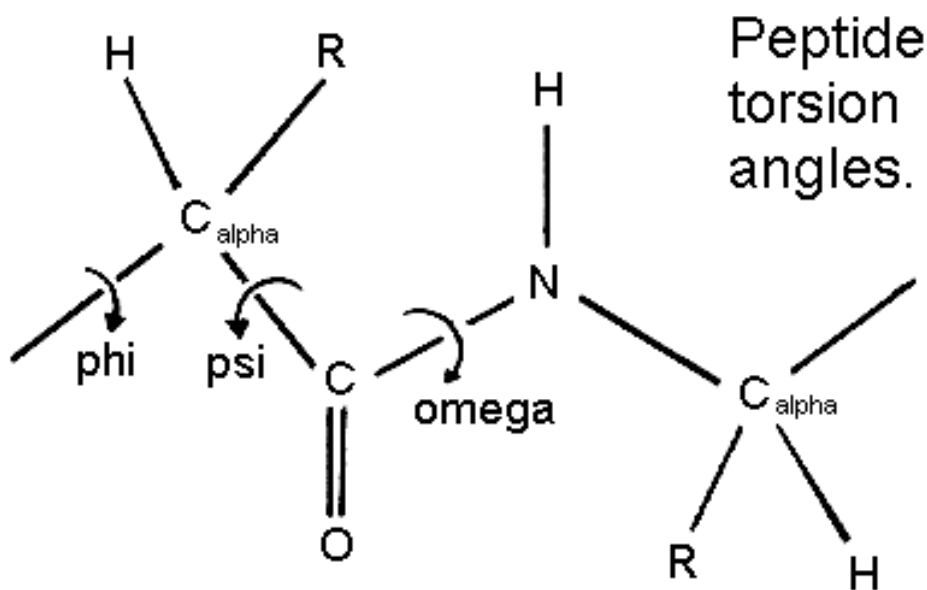
$$\vec{J}_{ia} = [\xi_{ia} \vec{e}_a \times (\vec{r}_i - \vec{s}_a)] + [\vec{\tau}_a + \vec{\mu}_a \times \vec{r}_i] \quad (2.6)$$



where the first expression in the first bracket is the transformation on the set of atoms  $\Omega_a$  and the expression in the second bracket is the correction added to fulfill the Eckart conditions. The Hessian matrix, or stiffness matrix, can be computed as

$$H_{ab}^{(\theta)} = \frac{\partial^2 U}{\partial \theta_a \partial \theta_b} = (J^T H^{(r)} J)_{ab} \quad (2.7)$$

Here,  $J \in \mathbb{R}^{3n \times 2n}$  is the Jacobian matrix,  $J^T$  is the transpose of  $J$ , and  $H^{(r)} \in \mathbb{R}^{3n \times 3n}$  denotes the Hessian matrix in Cartesian coordinates.



**Figure 2.** Torsion angles on the protein backbone. TNM defines phi and psi angles as the DoF while keeping omega fixed. (Figure adapted from the online course “The Principles of Protein Structure” organized by Birkbeck College [52])

Normal modes of this system are computed by solving the general eigen problem

$$(\mathbf{J}^T \mathbf{H}^{(r)} \mathbf{J}) \vec{u}^\alpha = \omega_\alpha^2 (\mathbf{J}^T \mathbf{M} \mathbf{J}) \vec{u}^\alpha \quad (2.8)$$

Solving for the tridiagonal matrix  $\mathbf{P}$  in the Cholesky decomposition  $\mathbf{J}^T \mathbf{H}^{(r)} \mathbf{J} = \mathbf{P}^T \mathbf{P}$  and replacing  $\vec{u}^\alpha$  by the mass-weighted normal modes  $\vec{w}^\alpha = \mathbf{L} \vec{u}^\alpha$ , (2.8) can be rewritten as a standard eigenvalue problem

$$\left( (\mathbf{J} \mathbf{P}^{-1})^T \mathbf{H}^{(r)} (\mathbf{J} \mathbf{P}^{-1}) \right) \vec{w}^\alpha = \omega_\alpha^2 \vec{w}^\alpha \quad (2.9)$$

The original eigenvectors that were sought after is recovered with  $\vec{u}^\alpha = \mathbf{P}^{-1} \vec{w}^\alpha$ , which, in Cartesian coordinates, are given by  $\vec{x}^\alpha = \mathbf{J} \vec{u}^\alpha$  and are orthonormal with respect to the mass tensor, i.e.,  $\sum_i m_i \vec{x}^\alpha \cdot \vec{x}^\beta = \delta_{\alpha\beta}$ . Since the equipartition theorem states that the kinetic energy is evenly distribution across DoFs, it implies that the contribution  $z^\alpha$  of mode  $\alpha$  to the thermal fluctuation is proportional to  $\omega_\alpha^{-2}$

$$z^\alpha \propto \sum_i m_i \langle (\vec{r}_i - \vec{r}_i^0)^2 \rangle = \left( \frac{k_B T}{\omega_\alpha^2} \right) \sum_i m_i |\vec{x}_i^\alpha|^2 \propto \omega_\alpha^{-2} \quad (2.10)$$

with  $k_B$  denoting the Boltzmann constant and  $T$  denoting the temperature.

Computation of TNM is done by with code provided by the authors of TNM on GitHub (<https://github.com/ugobas/tnm>).

A few dynamical couplings to make sense of and summarize the various modes were also proposed by the same group in later work [53]. By definition, these couplings can be computed for each pair of atoms in the protein. They are computed only between  $C_\alpha$  atoms in this work to characterize the relationship between pairs of residues. The co-directionality, coordination, and deformation



couplings are defined as follows.

## Co-Directionality Coupling

The co-directionality coupling measures the collinearity in movement between pairs of residues. It is closely related to the covariance matrix but with the effect of amplitude removed. Co-directionality coupling between the  $i$ -th and the  $j$ -th atom is given by

$$C_{ij}^{\text{codir}} = \frac{\sum_{\alpha} \frac{1}{\omega_{\alpha}^2} \frac{\vec{x}_i^{\alpha} \cdot \vec{x}_j^{\alpha}}{|\vec{x}_i^{\alpha}| |\vec{x}_j^{\alpha}|}}{\sum_{\alpha} \frac{1}{\omega_{\alpha}^2}} \quad (2.11)$$

Note that the various modes are weighted by its energetic contribution, which is derived based on the equipartition theorem.

Co-directionality is large if the pair of residues tend to move in either parallel or anti-parallel directions, and low if they tend to move perpendicularly.

## Coordination Coupling

Coordination coupling reflects the fluctuation of distance between two atoms and is defined as

$$C_{ij}^{\text{coord}} = 1 - 0.5 \sqrt{\sum_{\alpha} \frac{1}{\omega_{\alpha}^2} \left( \frac{\vec{r}_i^0 - \vec{r}_j^0}{r_{ij}^0} \cdot (\vec{x}_i^{\alpha} - \vec{x}_j^{\alpha}) \right)^2} \quad (2.12)$$

The parameters 1 and 0.5 are chosen specifically for TNM such that most of the



couplings are positive. Again, the summation takes the energetic contribution into consideration.

Pairs of residues with large coordination couplings tend to maintain the same distance under thermal equilibrium, and vice versa.

## Deformation Coupling

Deformation coupling quantifies how a unit perturbation force  $\vec{f}$  applied at the  $i$ -th residue induces deformation on the  $j$ -th atom. The direction of the force  $\vec{f}$  is defined such that it maximizes the deformation at the  $j$ -th atom:

$$|\vec{r}_i - \vec{r}_i^0|^2 = \vec{f} \cdot \left( (\mathbf{F}^{(ij)})^T \mathbf{F}^{(ij)} \vec{f} \right) \quad (2.13)$$

where  $\mathbf{F}^{(ij)}$  is a  $3 \times 3$  matrix with components  $F_{mn}^{(ij)} = \sum_{\alpha} x_{im}^{\alpha} x_{in}^{\alpha} / \omega_{\alpha}^2$ , with  $(x_{ix}^{\alpha}, x_{iy}^{\alpha}, x_{iz}^{\alpha})$  being the three components in the Cartesian eigenvector  $\vec{x}^{\alpha}$  corresponding to the  $i$ -th atom. The maximum value of the deformation is equal to the maximum eigenvalue of the matrix  $(\mathbf{F}^{(ij)})^T \mathbf{F}^{(ij)}$ , leading to

$$C_{ij}^{\text{deform}} = \max \left\{ \text{eigenval} \left( (\mathbf{F}^{(ij)})^T \mathbf{F}^{(ij)} \right) \right\} \quad (2.14)$$

Deformation coupling is large when a unit perturbation force on the  $i$ -th residue is able to induce a large deformation at the  $j$ -th residue, and small when movement at the  $j$ -th residue is insensitive to forces applied on the  $i$ -th residue.

## 2.2 Mapping Proteins to Graphs



This section deals with the representation of protein structure and dynamics as graphs. Before elucidating the mapping between proteins and graphs, some notation and definitions are given.

### 2.2.1 Graph Theory

**Definition.** A *graph*  $G$  is defined as a paired set  $G = (V, E)$ , where  $V$  can be any nonempty set, and  $E \subseteq \{\{v_1, v_2\}: (v_1, v_2) \in V \times V\}$ .  $V$  is the set of vertices, or nodes, of  $G$ , and elements of  $E$  are known as the *edges* of  $G$ .

A graph  $G$  can be categorized as *directed* or *undirected* based on whether the elements of  $E$  all fulfill specific criteria. A graph is directed if the elements  $\{v_i, v_j\} \in E$  are ordered pairs and undirected if not ordered. This work is focused solely on directed graphs.

**Definition.** The *neighborhood* of a vertex  $v$  is defined as  $N_v = \{w \in V: \{v, w\} \in E\}$ , or the set of vertices connected to  $v$ , possibly including  $v$  itself. Elements of  $N_v$  are known as the *neighbors* of  $v$ .

Edges that connect a node with itself, i.e.,  $\{v, v\}$ , are known as *self-loops*. It





follows that vertices with self-loops have themselves in their neighborhood.

Undirected graphs without self-loops are said to be *simple*.

**Definition.** The *degree* of a vertex  $v$ ,  $\deg(v)$ , is equal to the number of neighbors of  $v$ , i.e., the cardinality of  $N_v$ , or  $|N_v|$ .

The information of a graph can be succinctly stored or presented with the *adjacency matrix*  $A$  if some (arbitrary) order is imposed on the set of vertices  $V$ .

**Definition.** With the set of vertices of  $G$  ordered as  $\{v_1, v_2, \dots, v_{|V|}\}$ , the adjacency matrix  $A \in \mathbb{R}^{|V| \times |V|}$  is defined as  $A_{ij} = 1$  if  $\{v_i, v_j\} \in E$ , otherwise  $A_{ij} = 0$ .

It follows immediately that the adjacency matrix is necessarily symmetric for undirected graphs. Another standard matrix for representing the graph information of graphs is the *Laplacian matrix*  $L$ , which finds use in spectral graph theory and spectral-based graph neural networks, such as graph convolutional networks.

**Definition.** The *degree matrix*  $D \in \mathbb{R}^{|V| \times |V|}$  is a diagonal matrix where  $D_{ii} = \deg(v_i)$ . The Laplacian matrix  $L$  is equal to  $D - A$ .

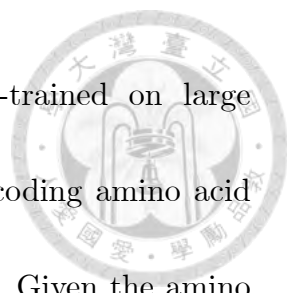
*Vertex descriptors* can be defined for each vertex. Suppose the descriptors are  $d$ -dimensional; then the descriptor for vertex  $v_i$  is denoted by the  $d$ -dimensional

vector  $\vec{x}_{v_i}$ . The entirety of descriptors for all vertices can be collected into a *vertex descriptor matrix*  $\mathbf{X} \in \mathbb{R}^{|V| \times d}$ .



## 2.2.2 The Data Representation

Ideally, the data representation of a protein should include the most critical, if not all, information required to make accurate predictions of all properties of the protein. Any attempt in achieving such an encoding would require the practitioner to determine what features of the protein are considered critical. In this work, it is assumed that the essence of protein properties lies in its sequence, structure, and dynamical behavior. These are encoded as either different types of graphs or as vertex descriptors. In particular, sequential information is encoded as a combination of vertex descriptors and the *backbone* graph. Structural information is stored in the *contact* graph and the *predicted alignment error* (PAE) graph. Dynamical information is summarized in three graphs: the *co-directionality* (codir) graph, the *coordination* (coord) graph, and the *deformation* (deform) graph. Six types of graphs are defined in total, all sharing the same set of vertices, namely the residues:  $V = \{v_1, v_2, \dots, v_n\}$  where  $v_i$  corresponds to the  $i$ -th residue in the sequence of length  $n$ . The only difference between these graphs lies in the edge connections.



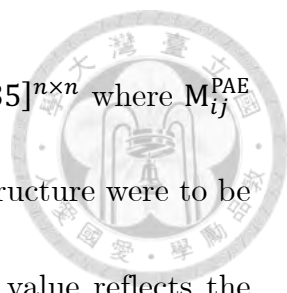
ProteinBERT is a bidirectional transformer model pre-trained on large databases for protein sequences and is commonly used for encoding amino acid sequences to facilitate interpretation for downstream processes. Given the amino acid sequence of length  $n$  as the input, the pre-trained model outputs a 1024-dimensional embedding vector for each residue plus two additional vectors of the same dimension for the "start" and "end" tokens. The two additional vectors are discarded in this work and the remaining vectors are collected as the vertex feature matrix  $X_{\text{seq}} \in \mathbb{R}^{n \times 1024}$ . This completes the sequential embedding features.

The three-dimensional structure for all proteins processed in this work is retrieved from the AlphaFold protein structure database and is the basis for computing both the protein contact map and TNM dynamical couplings. The AlphaFold-predicted structure assigns a position to each of the residues in the sequence. Denote the position of  $C_\alpha$  of the  $i$ -th residue by  $\vec{r}_{v_i}$ , a three-dimensional vector. Based on these position vectors, the protein contact map is built by adding an edge between any pair of residues whose  $C_\alpha$  atoms are located within a preset cutoff distance  $c_{\text{contact}}$ :

$$E_{\text{contact}} = \left\{ \{v_i, v_j\} : (v_i, v_j) \in V \text{ and } \left| \vec{r}_{v_i} - \vec{r}_{v_j} \right| < c_{\text{contact}} \right\} \quad (2.15)$$

The cutoff distance  $c_{\text{contact}}$  is set to 12 Angstroms in this work.

AlphaFold also generates the predicted alignment error (PAE) to accompany

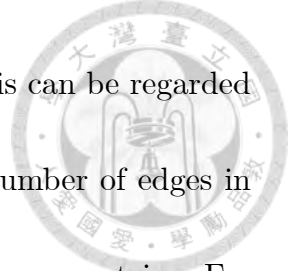


the structure prediction. It is provided as a matrix  $M^{\text{PAE}} \in [0,35]^{n \times n}$  where  $M_{ij}^{\text{PAE}}$  is the expected position error at  $j$ -th residue if the predicted structure were to be aligned with the true structure true at the  $i$ -th residue. This value reflects the model's confidence in the relative position of the pair and is especially useful in assessing domain packing. For example, PAE is generally low within domains where the structure is more rigid but generally high between domains connected by linkers or spacers. Although the PAE matrix is not necessarily symmetric, inspection reveals that the upper and lower triangular portion of the matrix are indeed very similar. In this study, the lower triangular portion is discarded and the upper triangular portion is mirrored to complete a symmetric matrix. Edges are defined for every pair with predefined threshold  $c_{\text{PAE}}$

$$E_{\text{PAE}} = \left\{ \{v_i, v_j\}: (v_i, v_j) \in V \text{ and } M_{ij}^{\text{PAE}} > c_{\text{PAE}} \text{ and } j \geq i \right\} \quad (2.16)$$

and an undirected PAE graph is thus generated from the PAE matrix. The threshold  $c_{\text{PAE}}$  is set to 4 Angstroms throughout this work. Both the contact graph and the PAE graph are considered structural information.

The three dynamical coupling from the previous subsection can be directly converted into graphs by defining a threshold such that an edge is generated for every pair of residues with coupling larger than the threshold. The definition of this threshold, however, is different from the contact graph or the PAE graph in



that it is not a constant but varies from protein to protein. This can be regarded as a normalization method to avoid large fluctuations in the number of edges in different proteins due to the wide range of coupling values across proteins. For each protein, the threshold is defined as the mean value of all couplings plus twice the standard deviation. Thus, the set of edges for the co-directionality graph is given by

$$E_{\text{codir}} = \left\{ \{v_i, v_j\}: (v_i, v_j) \in V \text{ and } C_{ij}^{\text{codir}} > c(C^{\text{codir}}) \right\} \quad (2.17)$$

where  $c(C^{\text{codir}}) = \mu(\{C_{mn}^{\text{codir}}: m < n\}) + 2 \times \sigma(\{C_{mn}^{\text{codir}}: m < n\})$ ,  $\mu$  is the mean function and  $\sigma$  is the standard deviation function. This notation where  $c$  takes  $C^{\text{codir}}$  as input highlights that the cutoff value is protein-dependent and not a constant. Similarly, the set of edges for the coordination graph is

$$E_{\text{coord}} = \left\{ \{v_i, v_j\}: (v_i, v_j) \in V \text{ and } C_{ij}^{\text{coord}} > c(C^{\text{coord}}) \right\} \quad (2.18)$$

and the set of edges for the deformation graph is

$$E_{\text{deform}} = \left\{ \{v_i, v_j\}: (v_i, v_j) \in V \text{ and } C_{ij}^{\text{deform}} > c(C^{\text{deform}}) \right\} \quad (2.19)$$

The formulation of dynamical graphs is now complete.

Two residue-level features remain in the graph representation of proteins: B-factors and predicted LDDT- $C_\alpha$  (pLDDT). B-factors, also known as the Debye-Waller factor in condensed matter physics, is an index that reflects the relative vibrational motion of each atom. It is commonly reported alongside protein structures solved by X-ray crystallography and is defined as

$$B = \frac{8\pi^2}{3}\langle u^2 \rangle \quad (2.20)$$

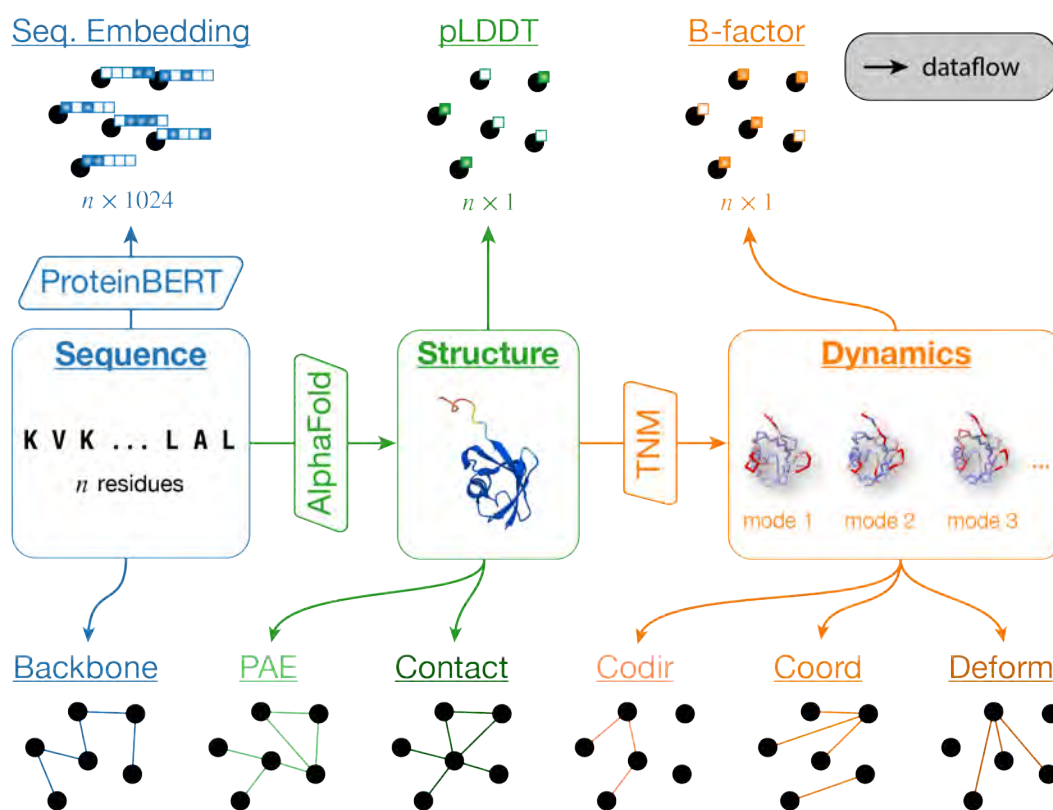
where  $\langle u^2 \rangle$  is the average movement squared measured in Angstrom squared ( $\text{\AA}^2$ ).

Since the protein structures used in this work are not experimentally resolved, values for B-factors are estimated from TNM analysis, which is reported per residue. On the other hand, pLDDT is generated by AlphaFold for every residue in the structure as a measure of the model's confidence in the local structure. The values range from 0 to 100, with high values usually found in highly structured regions such as regions of alpha helices. Lower values are found on linkers or spacers. While PAE reflects confidence in global features, pLDDT reflects confidence in local structure. Both B-factors and pLDDT are residue-wise, and hence serve as vertex descriptors. The full vertex feature matrix  $X$  is of size  $n \times 1026$  and composed of the ProteinBERT embedding, B-factor, and pLDDT.

A summary of the features defined for a protein is given in Table 1. The various graphs for a sample protein (accession number: A0A061ACL6) are displayed in Figure 4 as adjacency matrices.

Information	Type of Graph	Vertex Descriptors
Sequence	Backbone	ProteinBERT embedding
Structure	Contact Predicted alignment error (PAE)	Predicted LDDT (pLDDT)
Dynamics	Co-directionality (codir) Coordination (coord) Deformation (deform)	TNM-predicted B-factors

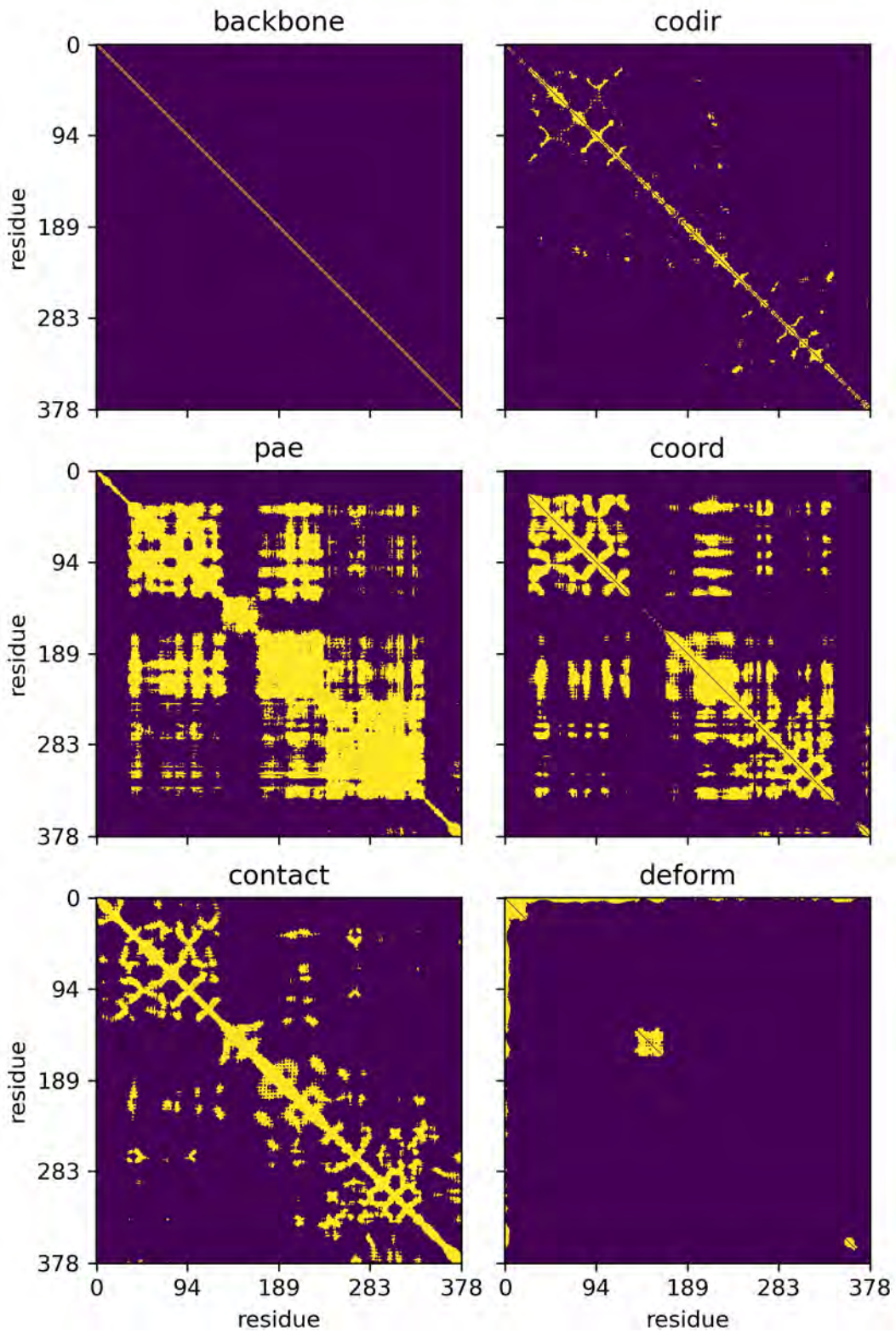
**Table 1.** Summary of the features for the graph representation of proteins.



**Figure 3.** Dataflow for building the graph representation of proteins. The structure is predicted by AlphaFold, and the dynamical analysis is also based on the AlphaFold structure. The backbone edges (blue) can be derived directly from the sequence. Predicted alignment error (PAE) edges and contact edges are considered structural information (green). Dynamical information (orange) is encoded in the co-directionality (codir), coordination (coord), and deformation (deform) edges.



A0A061ACL6-AFv4 (378 residues)



**Figure 4.** Showcase of the six types of graphs defined in this work, presented as adjacency matrices. Yellow indicates connections.



## 2.3 Data Source and Feature Distribution



This section explains the source of the data used in this work and some basic dataset analysis to uncover correlation between the features (graph representation) and the regression target (melting temperature).

### 2.3.1 Data Processing

Values of melting temperature used in this work are derived from the dataset provided by DeepSTABp [28], where the vast majority of data was collated from the meltome atlas [24]. The melting temperature of proteins from 12 different species is summarized in Table 2. The data were measured by thermal proteome profiling (TPP) and can be separated into two subsets by how proteome profiles were obtained: 1) by heating cells and 2) by heating lysates. To simplify the model, this work focuses solely on the prediction of lysate-based melting temperature. 29,758 proteins were included in the lysate-based subset, of which 9,503 entries were discarded due to either 1) an error during TNM execution or 2) computation taking longer than 20 seconds on an AMD R7 5800X processor. This resulted in 20,255 proteins with melting temperatures ranging from 30.4°C to 92.57°C with a



mean value of 50.43°C. The final dataset was randomly split by 8:2 into a training set and a validation set, and these datasets were used across all experiments in this work. No test sets were constructed in this work.

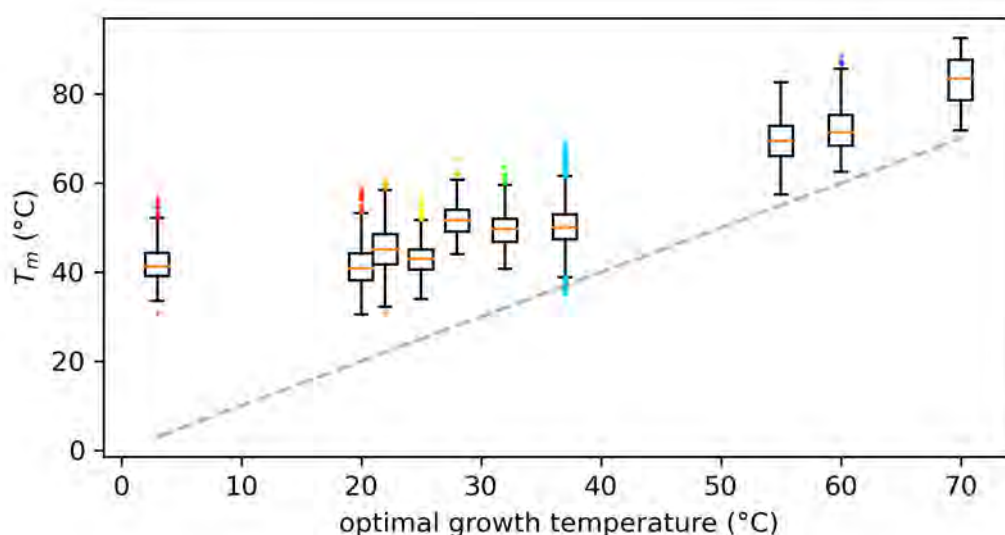
Organism	$T_{og}$	Mean of $T_m$	StD of $T_m$	# Entries
Oleispira antarctica	3	42.0	3.9	886
Caenorhabditis elegans	20	41.4	4.5	2,063
Arabidopsis thaliana	22	45.1	4.9	1,594
Drosophila melanogaster	25	43.0	3.4	907
Danio rerio	28	52.1	4.3	118
Saccharomyces cerevisiae	32	49.4	4.0	1,430
Bacillus subtilis	37	44.3	3.9	1,173
Escherichia coli	37	50.3	5.8	1,386
Homo sapiens	37	51.4	3.9	4,620
Mus musculus	37	50.4	3.7	4,188
Geobacillus stearothermophilus	55	69.6	5.0	555
Picrophilus torridus	60	72.0	4.8	755
Thermus thermophilus	70	82.9	5.5	580
<b>Total</b>				20,255

**Table 2.** The 12 species included in the dataset, their optimal growth temperature  $T_{og}$ , and the number of proteins. Around 20% of the entries are from humans, and about 10% are from species with  $T_{og}$  greater than 50°C.

The correlation between  $T_{og}$  and  $T_m$  is plotted in Figure 5. It has been reported in the literature that the melting temperature is highly correlated with the optimal growth temperature  $T_{og}$  [22]. Although  $T_{og}$  benefits performance, its use is questionable when estimating the melting temperature of designed proteins.




It could be argued that  $T_{og}$  is a reference point for the model to access and memorize the environment in which the proteins are synthesized, such as the effect of available post-translation modification specific to each species or the different chaperones specific to each species. If this is the case, using  $T_{og}$  as a feature would make sense when the designed protein is expressed in a particular cell line. However, this also means that it would not make sense to use  $T_{og}$  as a feature if the protein is artificially synthesized.



**Figure 5.** A significant correlation exists between the optimal growth temperature  $T_{og}$  and melting temperature  $T_m$ . The dashed line marks  $T_{og} = T_m$ .

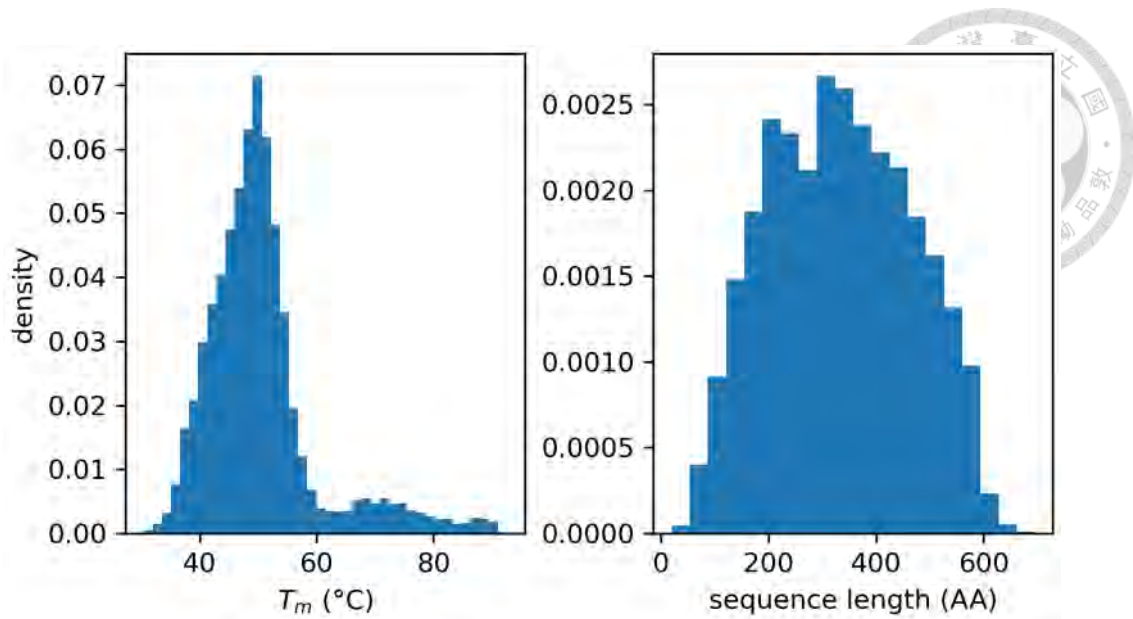
Whether  $T_{og}$  is reflective of the environment of synthesis remains to be seen. Although expressing the same sequence indifferent cells would likely result in chemically different molecules, their difference in  $T_m$  might not be significant. It would only make sense to reject  $T_{og}$  as a feature, and it should be possible to



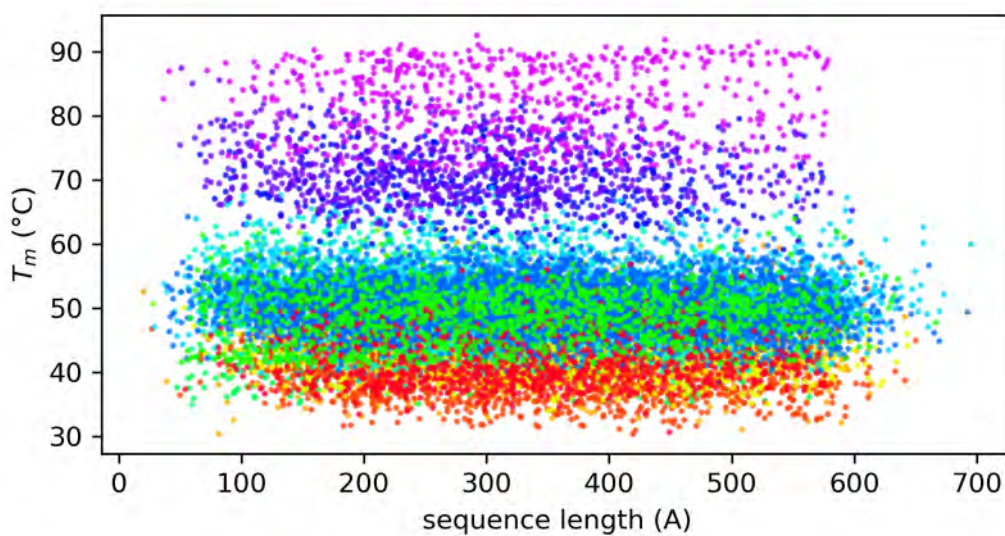
develop a model that can accurately predict  $T_m$  from the sequence alone. However, this cannot be proved with currently available datasets because all available entries are native to the species in which it was synthesized.  $T_m$  will be used as an input feature in this work.

### 2.3.2 Feature Analysis

The distribution of  $T_m$  is shown in Figure 6. Due to the inclusion of the thermophilic bacteria *Thermus thermophilus*, a long tail can be observed at higher temperatures. The mean value of  $T_m$  is 49.8242. The distribution of sequence length is also shown in Figure 6. The sequence length distribution is less skewed than  $T_m$  and ranges from 20 to 695, with an average length of 333.8 and a standard deviation of 132.6. Plotting  $T_m$  against sequence length in Figure 7, it is found that the correlation between the two is generally weak.



**Figure 6.** The distribution of melting temperature  $T_m$  and sequence length of all the proteins in the dataset used throughout this work. The melting temperature  $T_m$  is skewed towards higher temperatures, partly due to the inclusion of the thermophilic bacteria *Thermus thermophilus*.  $T_m$  is ranged between 30.4°C and 92.6°C, with mean, standard deviation, and kurtosis of 50.4, 9.7, and 3.4, respectively. The sequence length is ranged between 20 and 695, with an average of 333.8 and a standard deviation of 132.6.

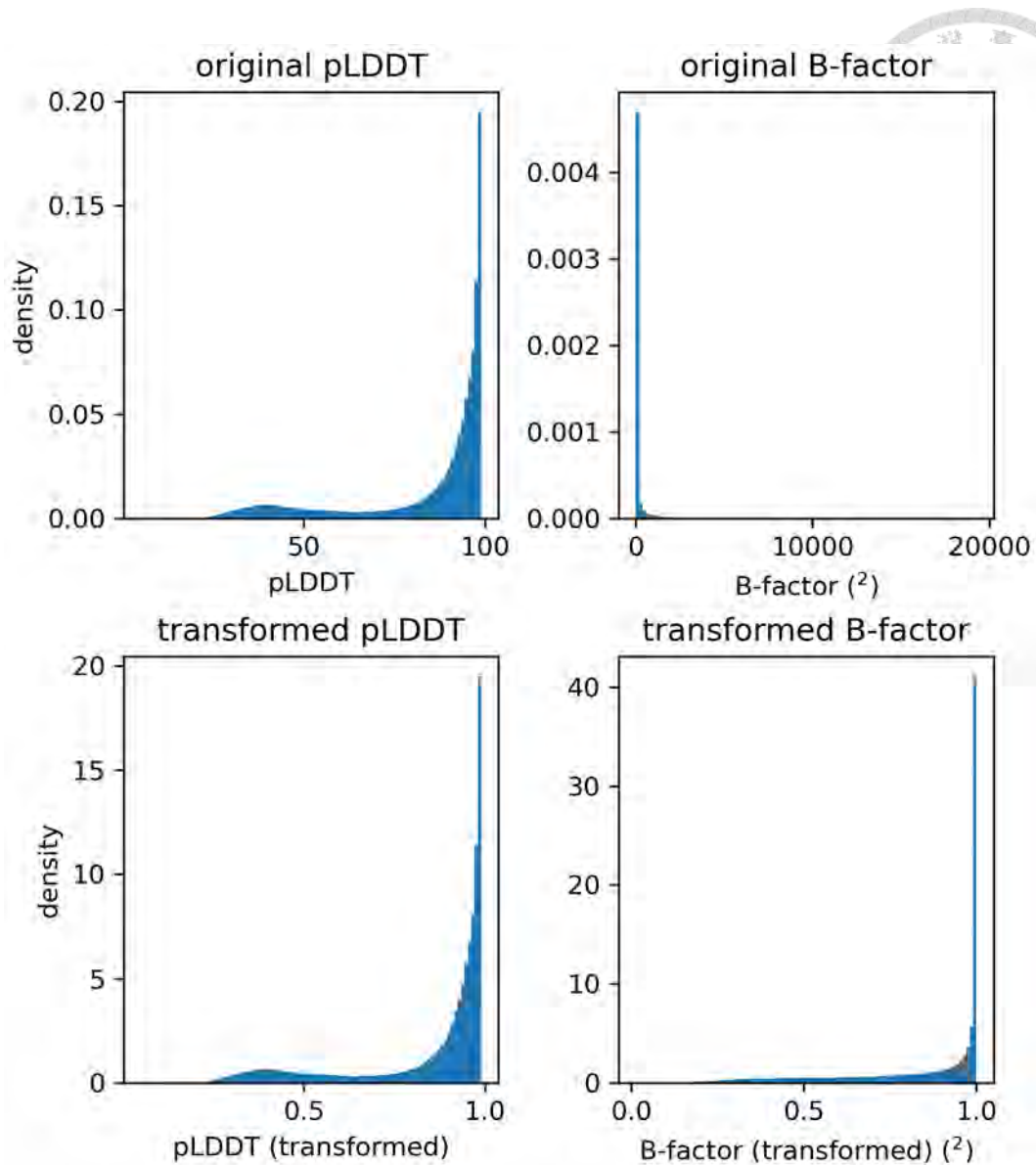


**Figure 7.** No significant correlation is found between sequence length and melting temperature  $T_m$ . (Each color corresponds to one of the 12 species in

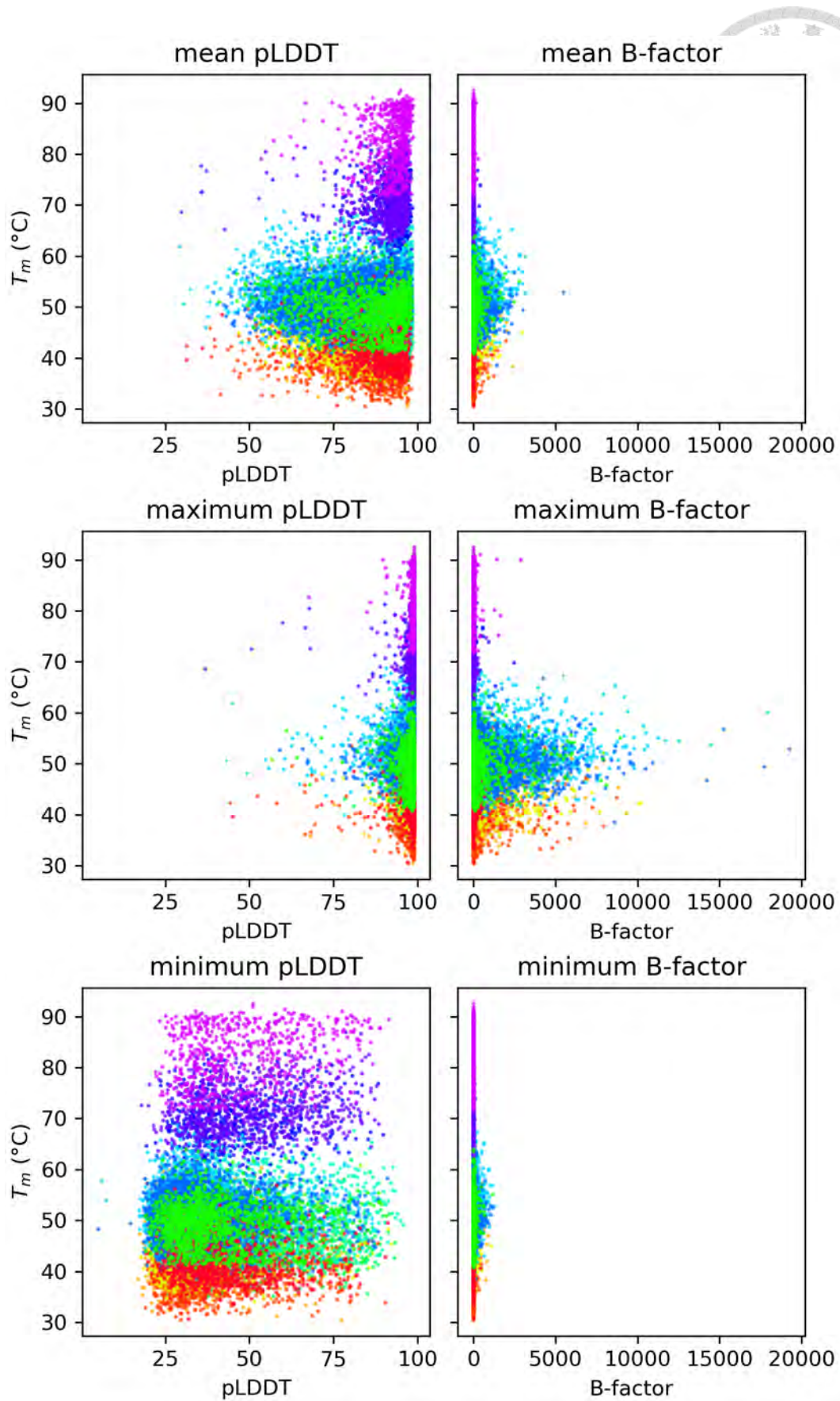
the dataset.)



pLDDT from AlphaFold and B-factor computed from the TNM analysis are both reported residue-wise. Their distribution within the entire dataset is shown in the first row of Figure 8. The second row shows these values after normalization/transformation.  $T_m$  is plotted against the protein-wise mean, maximum, and minimum value of pLDDT/B-factor in Figure 9, where it is shown that pLDDT for proteins with higher  $T_m$  are generally higher, meaning that AlphaFold has more confidence in the predicted structure. High pLDDT usually occurs in domains with packed secondary structures; therefore, there is some correlation between  $T_m$  and how well a protein is packed. On the other hand, smaller B-factors are found in proteins with higher  $T_m$  as well, reflecting a similar correlation as pLDDT.

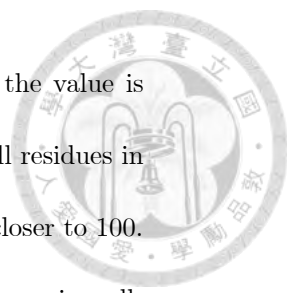


**Figure 8.** Distribution of pLDDT and B-factor of all residues from all proteins in the dataset. The first row shows the raw values reported by AlphaFold or as computed by TNM. The second row gives the values after normalization and transformation, which is done to facilitate faster convergence of the neural network model. For pLDDT, the values are scaled by  $1/100$  such that the values fall between 0 and 1. Values for B-factors  $B$  are transformed by  $1 - \log(B + 1)/10$  and then clipped between 0 and 1.



**Figure 9.** Correlation between B-factor/pLDDT and the melting temperature

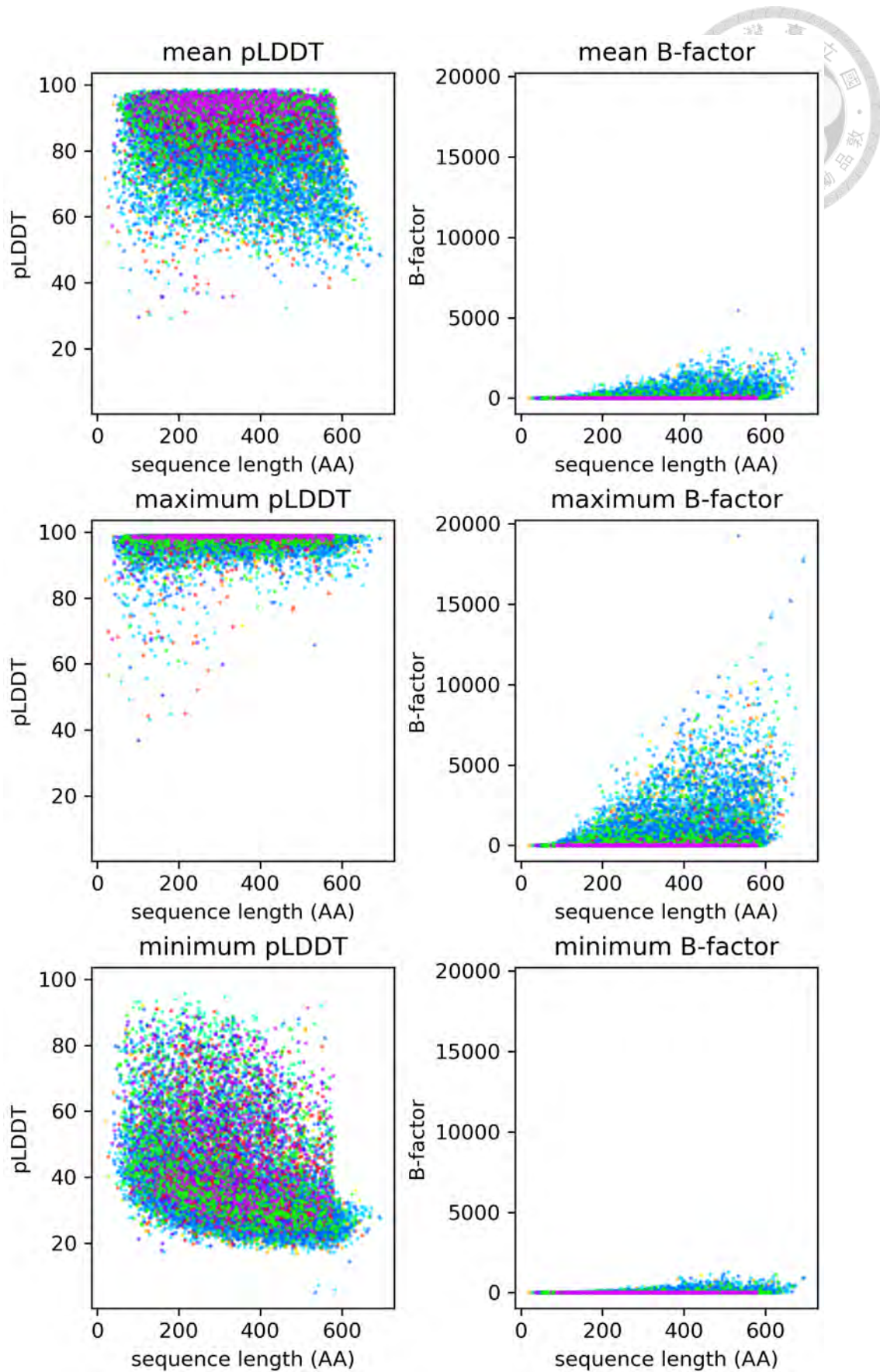




$T_m$ . Each point in the plot corresponds to a single protein, and the value is either the mean, maximum, or minimum value computed across all residues in the protein. Proteins with higher  $T_m$  have mean pLDDTs packed closer to 100. Higher pLDDT in the predicted structure of AlphaFold usually occurs in well-packed domains with a defined secondary structure, reflecting a correlation between higher thermostability and protein structure. The same correlation can be found between the B-factors. Note that the B-factors are computed from the AlphaFold structures in this figure. (Each color corresponds to one of the 12 species in the dataset.)

pLDDT and B-factors are plotted against the sequence length in [Figure 10](#).

The range of pLDDT slightly changes as the sequence length increases. A falloff of mean pLDDT across the dataset can be observed near the upper end of the sequence length, possibly due to the 20-second execution timeout during data processing. For B-factors, the range of maximum values increases with sequence length, although the mean values did not change as much.



**Figure 10.** Correlation between sequence length and pLDDT/B-factors. Each

point corresponds to a single protein in the dataset. (Each color corresponds to one of the 12 species in the dataset.)



## 2.4 The Neural Network Model

This section describes the architecture of the neural network models used throughout this work. The mechanism of the graph convolutional network and graph attention network is given in 2.4.1, followed by the architecture in 2.4.2.

### 2.4.1 Graph Neural Networks

Graph convolutional are employed for extracting graph features in this work. The goal of applying graph convolution is to learn high-level node embeddings on graph  $G = (V, E)$ . The convolution takes as input the ordered set of vertex features  $\{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_{|V|}\}$ , where each of the feature  $\vec{x}_i$  is a  $d_{in}$ -dimensional vector and outputs node embeddings as an ordered set of  $d_{out}$ -dimensional vectors. While the input features are summarized as the vertex feature matrix  $X \in \mathbb{R}^{|V| \times d_{in}}$ , the embeddings are summarized as an embedding matrix  $Y \in \mathbb{R}^{|V| \times d_{out}}$ . The rationale and computation of a specific type of graph convolution, named “graph convolutional network” (GCN), follows next.



# Graph Convolutional Network

Here, GCN refers to the layer-wise propagation rule for neural network models proposed in 2016 by Kipf and Welling for semi-supervised classification of nodes [54], as opposed to being an umbrella term referring to any graph neural network that makes use of localized filters/kernels. GCN became one of the most popular graph learning layers since its publication, largely due to its simplicity. Despite being designed for node-level tasks, it has seen extensive use in both edge- and graph-level predictions.

The layer-wise propagation rule of GCN for  $l$ -th layer is

$$\mathbf{H}^{l+1} = \text{act} \left( \tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{H}^l \mathbf{W}^l \right) \quad (2.21)$$

where  $\mathbf{H}^l \in \mathbb{R}^{|\mathcal{V}| \times d_l}$  denotes the matrix of vertex embedding on the  $l$ -th layer such that  $\mathbf{H}^0 = \mathbf{X}$ . Furthermore,  $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_{|\mathcal{V}|}$  is the adjacency matrix  $\mathbf{A}$  with added self-loop, and  $\tilde{\mathbf{D}}$  is the diagonal matrix with  $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}$ . Finally,  $\mathbf{W}^l \in \mathbb{R}^{d_l \times d_{l+1}}$  is the layer-specific trainable weight matrix, and  $\text{act}$  denotes the nonlinear activation function, e.g., ReLU or Sigmoid.

From a spectral point-of-view, GCN is a crude approximation of ChebNet [55], the simplification of spectral filters  $g_\theta$  by expanding the filter by Chebyshev polynomials  $T_k(x)$  up to the  $K$ -th order:

$$g_\theta \star \vec{s} \approx \sum_{k=0}^K \theta_k T_k(\tilde{L}) \vec{s} \quad (2.22)$$

where  $\star$  is the convolution operator,  $\vec{s} \in \mathbb{R}^{|V|}$  is the graph signal,  $\theta_k$  is the coefficient of each Chebyshev polynomial to be learned during training, and  $\tilde{L} = \frac{2}{\lambda_{max}} L_{sym} - I_{|V|}$  with  $L_{sym} = I_{|V|} - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$  being the symmetrically normalized Laplacian matrix. In other words, ChebNet approximates spectral convolution on graphs by a  $K$ -localized algorithm, i.e., the approximation only depends on vertices that are at most  $K$  neighbors apart. To recover GCN from (2.22), set  $K$  to 1 and approximate  $\lambda_{max}$  by 2, and the equation is further simplified to

$$g_\theta \star \vec{s} \approx \theta_0 \vec{s} - \theta_1 (D^{-\frac{1}{2}} A D^{-\frac{1}{2}}) \vec{s} \quad (2.23)$$

Then set  $\theta = \theta_0 = -\theta_1$  to reduce the number of parameters and, accordingly, the chance of overfitting, resulting in

$$g_\theta \star \vec{s} \approx \theta (I_{|V|} + D^{-\frac{1}{2}} A D^{-\frac{1}{2}}) \vec{s} \quad (2.24)$$

This expression is numerically unstable if multiple layers are stacked because the eigenvalues of  $I_{|V|} + D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$  fall in the range of  $[0, 2]$ . GCN's solution is to replace it with a formula of a similar form in a technique known as the *renormalization trick*:

$$I_{|V|} + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \rightarrow \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} \quad (3.25)$$

where  $\tilde{A} = A + I_{|V|}$  and  $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$ . In essence, the adjacency of the original graph structure is modified by the addition of self-loops. (3.21) is now recovered but without the activation function.



The propagation rule can be written in node-wise vector form as

$$\vec{h}_i^l = \text{act} \left( \sum_{v_j \in N_{v_i} \cup v_i} \frac{1}{c_{ij}} \vec{h}_j^l W^l \right) \quad (3.26)$$

where  $\vec{h}_i^l$  represents the  $d_l$ -dimensional descriptor of vertex  $v_i$  and  $c_{ij} = \sqrt{\tilde{D}_{ii} \tilde{D}_{jj}}$  is the normalization constant for edge  $\{v_i, v_j\}$  derived from  $\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$ . This formulation highlights the existence of the normalization coefficient such that the descriptors are stable with respect to the size of the number of neighbors of each vertex. It also highlights the importance of adding self-loops: without them, vertices "forget" their previous values on every propagation.

## 2.4.2 Architecture

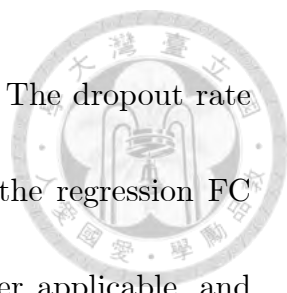
Since an objective of this work is to incorporate information on protein dynamics into the prediction of  $T_m$ , two models, OGT and DYN, are designed and compared in this work, as listed in Table 3. OGT is the baseline model, where the only feature given to the model is the optimal growth temperature  $T_{og}$ . This is compared against DYN, the model that has access to all the sequential, structural, and dynamical features as defined above. Both models are trained with the same training/validation split.

Model	Sequential	Structural	Dynamical	Optimal Growth Temp.	Num. of Parameters
OGT	X	X	X	O	359
SEQ	O	X	X	O	34,823
STR	O	O	X	O	109,063
DYN	O	O	O	O	234,151

**Table 3.** Overview of different models trained. OGT is the baseline model.

DYN is the focus of this work, where all the features are given to the model for learning.

The architecture of DYN is illustrated in Figure 11. The six different graphs are processed by six separate GCNs, where the  $n \times 1026$  vertex feature matrix is processed to produce an  $n \times 32$  vertex embedding matrix. The six embedding matrices corresponding to six different aspects of the protein are concatenated and passed through a global pooling layer to form the 192-dimensional *graph-level embedding*. On the other hand, the scalar  $T_{og}$  is passed through two fully connected (FC) layers of 20 and 10 neurons, respectively. The graph-level embedding is concatenated with this 10-dimensional  $T_{og}$  *embedding* to form a 202-dimensional *protein embedding* before being fed into the final three FC layers for regression. The number of neurons in each layer is designed to be linearly decreased from 202 to 1; thus, these FC layers have 134, 67, and 1 neurons, respectively. All the FC layers in the model, except for the output layer, contain the following components (in order): linear layer, dropout, and activation. The

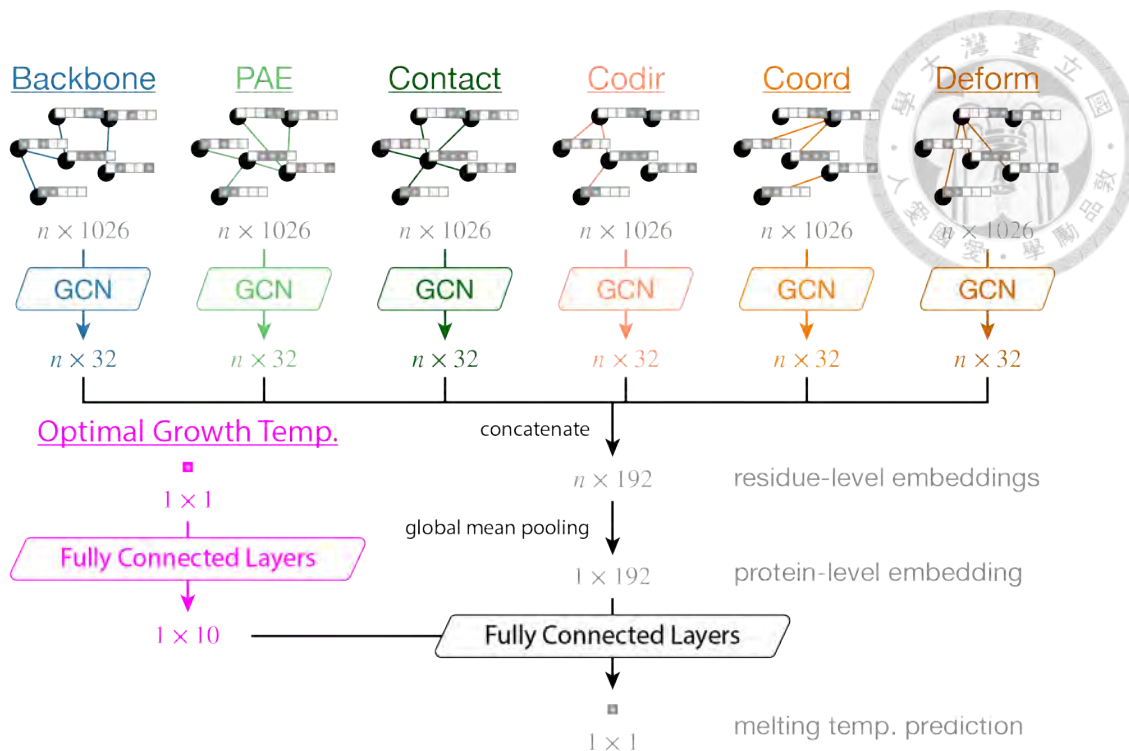


output layer consists of only a linear layer without activation. The dropout rate is set to 0.2 for the two  $T_{og}$ -processing FC layers and 0.5 in the regression FC layers. Leaky ReLUs are used for activation functions wherever applicable, and batch normalization with learnable affine transformation is added between the three FC layers in the final layers. This completes the architecture of DYN with 234,151 tunable parameters.

SEQ and STR can be obtained by removing unnecessary pipelines from DYN. For SEQ, only the pipeline for processing the backbone graph is kept. As a result, only 34,823 parameters remain for SEQ. The backbone, PAE, and contact pipelines are kept while dynamical pipelines are removed for STR. This model has 109,063 tunable parameters.

For OGT,  $T_{og}$  is also sent through an embedding block, i.e., the 20–10 neuron FC layer introduced in DYN. The  $T_{og}$  *embedding* is sent through three FC layers of the same design as in DYN, except for the number of neurons. In this case, the FC layers have 6, 3, and 1 neuron, respectively. The placement of the dropout, activation, and batch normalization layers is identical to DYN, as is the setup.

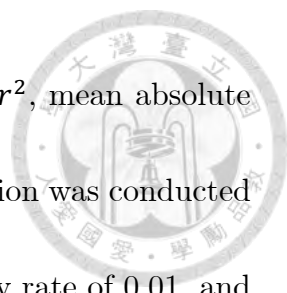




**Figure 11.** The neural network architecture of the DYN model trained in this study. The six types of graphs determine six different pathways along which vertex features are exchanged, leading to six distinctive vertex embeddings. These are concatenated and global pooled, and the protein-level embedding is obtained. Along with the 10-dimensional  $T_{og}$  embedding of the host, the 192-dimensional embedding is input into a set of fully connected layers to produce the desired  $T_m$  prediction. The OGT model is obtained with the protein-level embedding removed, while the SEQ and STR models are obtained by removing unnecessary pipelines from the architecture.

### 2.4.3 Training Setup

During training and prediction, the melting temperatures were standardized by the mean and standard deviation of the training set. The models were trained to minimize the mean squared error (MSE), and evaluated on the validation set



using a combination of Pearson correlation coefficient (PCC),  $r^2$ , mean absolute error (MAE), and root mean squared error (RMSE). Optimization was conducted using AdamW with an initial learning rate of 0.01, weight decay rate of 0.01, and  $\beta_1$  and  $\beta_2$  set to 0.9 and 0.999, respectively. The models were trained for 150 epochs with a batch size of 64. All computation was conducted on an HPC server (CPU: Intel Xeon Gold 5218, GPU: Nvidia V100). The code for machine learning was implemented using the PyTorch deep learning library [56] and the PyTorch Geometric graph learning library [57].

## 2.5 Regression Activation Map (RAM)

In 2016, the gradient class activation map (CAM) was proposed for convolutional neural networks as a technique to highlight the regions within the input image that the classifier deems most influential to the prediction [58]. In this work, a similar concept is applied on the regression model for  $T_m$  to aid recognition of regions in the protein that has the greatest influence on the overall thermostability (i.e.,  $T_m$ ) of the protein. A value, named the regression activation map (RAM), is to be attributed to each residue in the protein.

The concept of Grad-CAM, or in this case Grad-RAM, is straightforward: the extent to which the predicted value  $y$  depends on the vertex embedding

matrix  $\mathbf{H} \in \mathbb{R}^{n \times d}$  can be quantified by differentiating  $y$  with respect to all elements in  $\mathbf{H}$  and computing the average

$$w_k = \frac{1}{n} \sum_{i=1}^n \frac{\partial y}{\partial H_{ik}} \quad (3.27)$$

where  $w_k$  denotes the importance of the  $k$ -th dimension in the  $d$ -dimensional vertex embedding vector. RAM is computed as the weighted sum of all the layers (1 to  $d$ ) in  $\mathbf{H}$  by their respective importance. For the  $i$ -th residue, this is written as

$$s_i^{\text{RAM}} = \sum_k w_k H_{ik} \quad (3.28)$$

Residues with positive RAM value contribute positively to the prediction of the model, while residues with negative values have a negative impact on the prediction.



# Chapter 3.

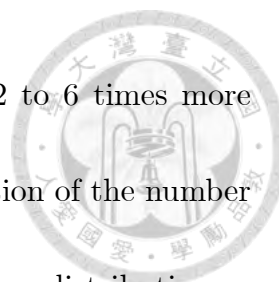


## Results and Discussion

### 3.1 The Graph Representation of Proteins

This section covers the results of data processing. Statistics regarding the different types of edges are reported, and their correlation with the melting temperature  $T_m$  is discussed.

Table 4 lists the minimum, maximum, mean value, and standard deviation of the number of edges within each protein in the dataset. The number of backbone edges is exactly equal to  $n - 1$ , where  $n$  is the sequence length. Contact, co-directionality, coordination, and deformation edges scale approximately proportionally to the volume of the protein. The number of these types of edges are of the same order of magnitude. On the other hand, the number of PAE edges is dependent on the interaction between protein domains. Therefore, it is unclear how it scales with sequence length. The order of magnitude of the maximum number of PAE edges is one order greater than the contact or the dynamical edges.



On average, the number of co-directionality edges is around 2 to 6 times more than contact, coordination, or deformation edges. The distribution of the number of edges is shown in Figure 12. The backbone edge has the same distribution as sequence length, while the other edge types all have a distribution skewed towards the right.

Edge Type	Minimum	Maximum	Mean	StD
Backbone	19	694	332.8	132.6
PAE	42	157,216	30,652.1	26,751.2
Contact	123	18,955	7,356.8	3,718.5
Co-directionality	2	84,523	12,045.1	13,231.9
Coordination	1	33,968	2,816.4	3,227.0
Deformation	11	13,127	2,051.9	1,628.9

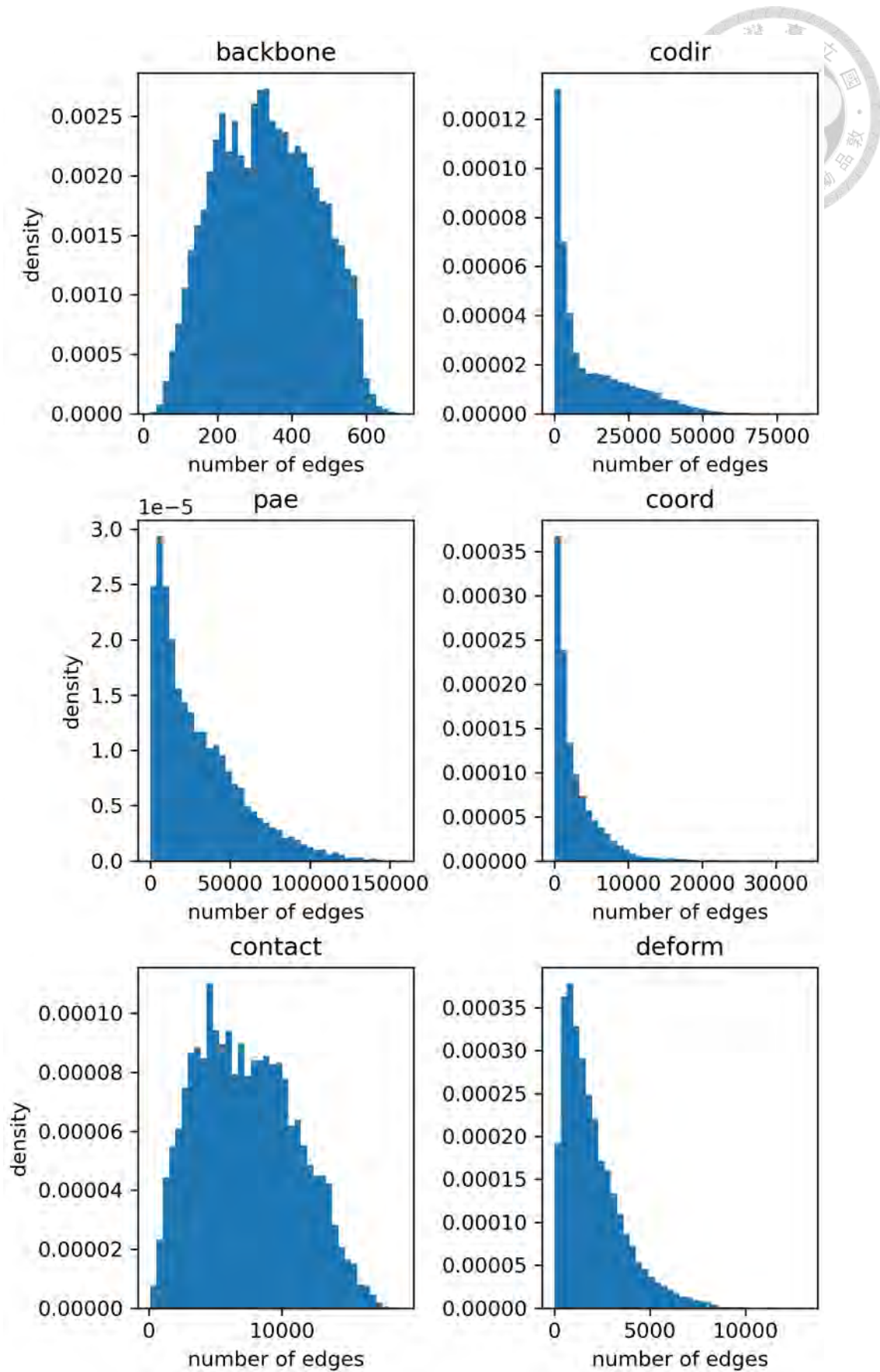
**Table 4.** Statistics on the various edge types.

The number of edges is plotted against the melting temperature  $T_m$  in Figure 13. While the backbone and the contact edges do not significantly correlate with  $T_m$ , the number of dynamical edges and PAE edges are smaller for proteins with larger  $T_m$ . However, proteins at the lower end of the  $T_m$  also had fewer edges when compared to proteins with  $T_m$  around 40 to 50°C.

The number of edges is plotted against sequence length in Figure 14. The range of the number of edges becomes larger as the sequence length increases, except for backbone length, where the number of edges is an exact function of

sequence length. A split in range can be found in both co-directionality and coordination edges and is especially significant in the former. However, further analysis is required to explain this phenomenon.





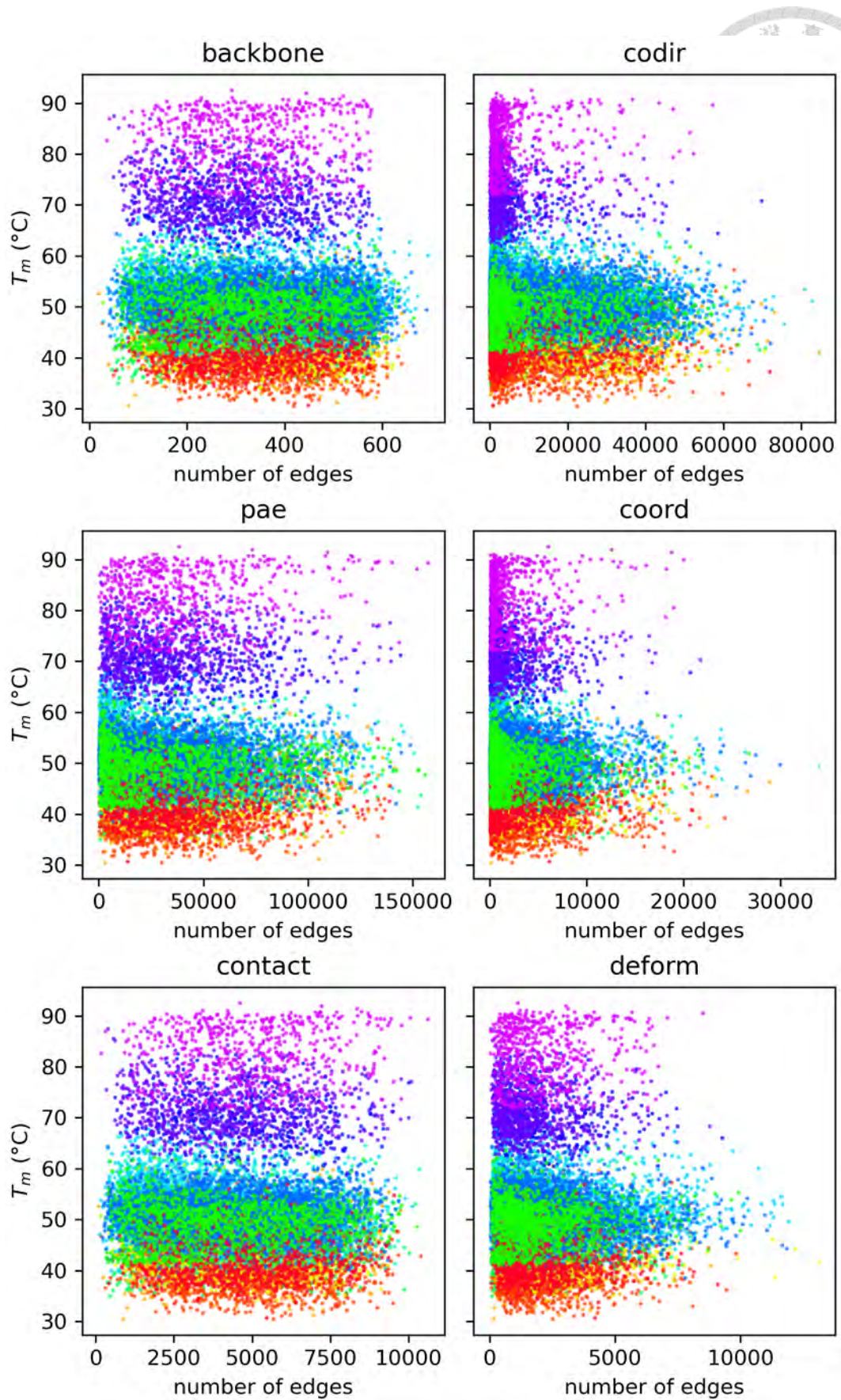
**Figure 12.** Distribution of the number of edges for all edge types. Backbone

edges have the same distribution as the sequence length distribution shown in

Figure 6.

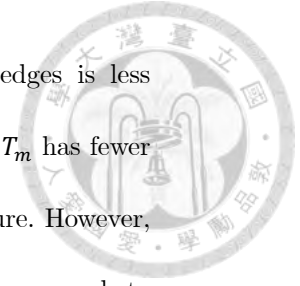


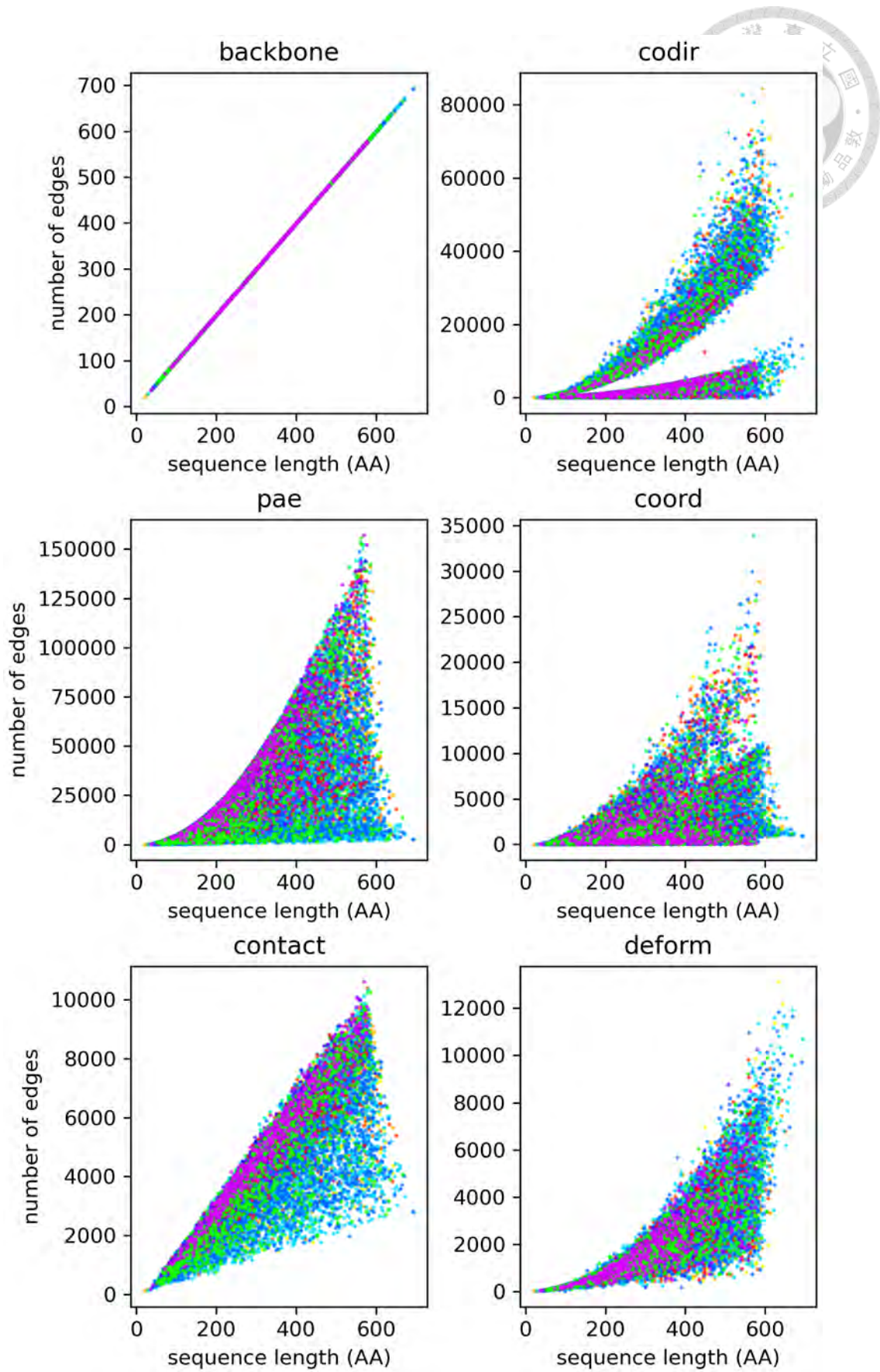




**Figure 13.** Relationship between the number of edges in each protein and their

$T_m$ . Correlation between the backbone edge and the contact edges is less significant than the dynamical edges, where proteins with higher  $T_m$  has fewer edges than proteins with  $T_m$  slightly higher than room temperature. However, the same can be observed for proteins with low  $T_m$ . (Each color corresponds to one of the 12 species in the dataset.)





**Figure 14.** Correlation between sequence length and the number of edges in

the various edge types. The lower and upper bound for all edge types increases as sequence length increases. A significant branching can be observed for co-directionality edges, while a less obvious branching is found for coordination edges. The reason for the branching remains to be uncovered. (Each color corresponds to one of the 12 species in the dataset.)




In summary, comparing the number of graph edges and  $T_m$  shows that the correlation between mere edge counts is not straightforward. Methods must look deeper than mere occurrences to learn the relation between graph edges and  $T_m$ .

## 3.2 Model Performance

The four models are trained with the data representation defined in 2.4.2 and the training setup described in 2.4.3. This section reports and discusses the performance and interpretability of the models.

### Graph Representation is Effective in $T_m$ Prediction

A summary of the performance of the four different models is given in Table 5, where five metrics are reported, namely mean absolute error (MAE), root mean squared error (RMSE), mean squared error (MSE), Pearson correlation coefficient (PCC), and the coefficient of determination ( $R^2$ ). Comparison between OGT and the other three models reveals that the graph representation proposed in this work

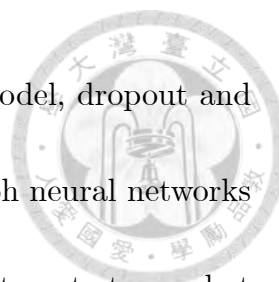


is effective in that the GNN model is able to extract useful information from this representation to improve performance by approximately 1.2°C. Since the difference between SEQ and STR is the use of structural graphs, it can be concluded that the structural graphs benefit the prediction despite only slightly increasing performance. It can also be concluded that dynamical graphs slightly aid prediction by comparing the results of STR and DYN in the same manner. In summary, the graph representation has successfully captured fundamental information about the protein. It should be noted that the number of parameters also increases as more features are utilized in each model, as was shown in Table 3.

Model	MAE	RMSE	MSE	PCC	$R^2$
OGT	4.433	5.565	30.969	0.859	0.671
SEQ	3.345	4.304	18.525	0.899	0.803
STR	3.307	4.290	18.405	0.901	0.805
DYN	3.291	4.286	18.373	0.901	0.805

**Table 5.** Performance of the four models.

The learning curve of the four models is given in Figure 15. The gap between training and validation gradually closes as more information is included as data features, and overfitting also gradually becomes significant. Combatting overfitting in DYN, however, is not a trivial task because graph neural networks



are notorious for overfitting. In the course of developing the model, dropout and dropout edges were gradually added, and the depth of the graph neural networks lessened to reduce overfitting, but to little avail. Numerous attempts to combat overfitting exist in the literature, each with specific strengths, weaknesses, and caveats, but no conclusive or unified treatment has been reached. Further tuning of the model is left as a future work.

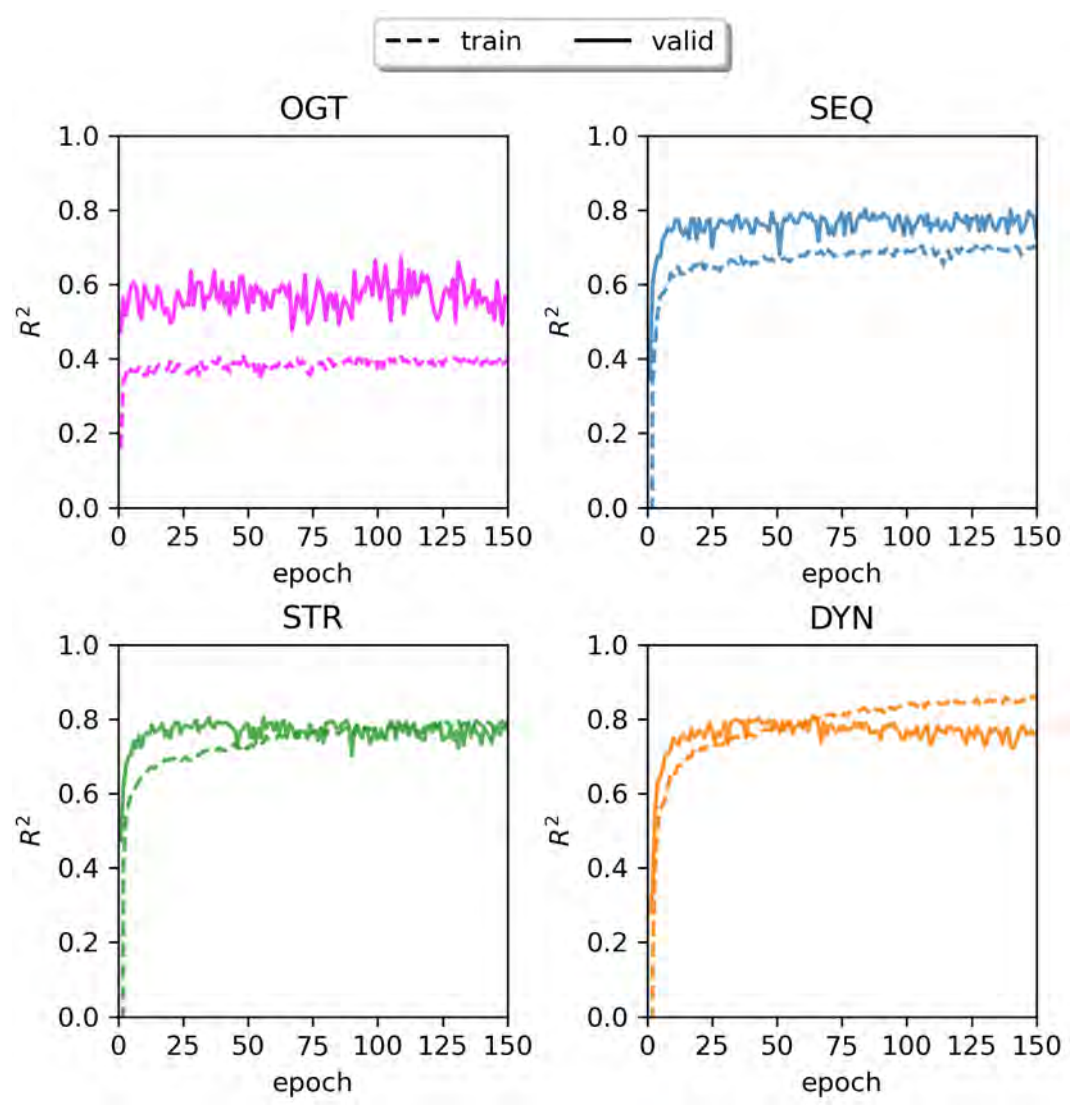


Figure 15. Learning curve ( $R^2$ ) of the four models trained in this work. The

gap between training and validation gradually decreases as more features are included as features, and overfitting gradually becomes significant. Overfitting is especially obvious in the DYN model, where all six types of graphs are used as input.



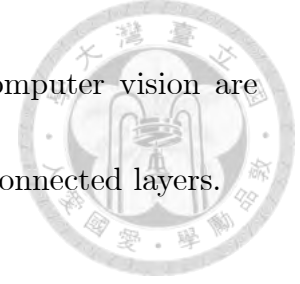
## Performance is Similar to the State-of-the-Art

### Method

A comparison of DYN with DeepSTABp, the state-of-the-art method for melting temperature prediction, is shown in Table 6. Values for DeepSTABp are reported by its authors in their publication [28]. The data distribution of the dataset used in DYN and DeepSTABp is expected to be similar because the dataset of DYN is derived from DeepSTABp, and the major difference lies in the number of proteins in the dataset. A comparison shows that DYN outperforms DeepSTABp in terms of RMSE, MSE, and  $R^2$  by a small margin and falls behind in terms of MAE, suggesting that the performance of DYN is very close to the state-of-the-art model.

Despite the similar performance, DeepSTABp has 28x more tunable parameters. Since DeepSTABp is composed entirely of fully connected layers, it is suspected that the model efficiency of DYN is bestowed by the use of graph

convolutional layers, similar to how image convolutions in computer vision are much more effective than simply feeding the image into fully connected layers.



Model	MAE	RMSE	MSE	PCC	$R^2$
DYN	3.291	4.286	18.373	0.901	0.805
DeepSTABp	3.22	4.30	18.46	0.90	0.80

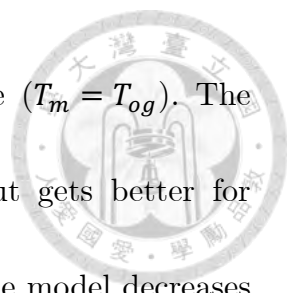
**Table 6.** Comparison of performance between DYN and the state-of-the-art model, DeepSTABp. Both models show similar performance. (Metrics for DeepSTABp are reported in its publication [28].)

## Bias and Variance of the Model

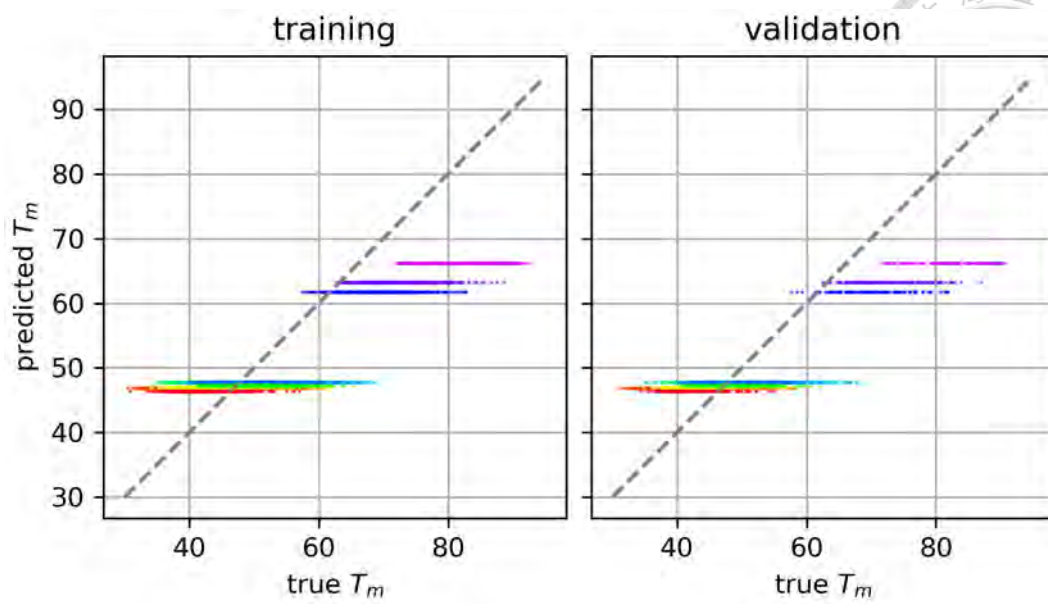
The predicted  $T_m$  of OGT is plotted against its true values for proteins in the training and validation datasets in Figure 16. The predicted values are stratified because  $T_{og}$  is the only input feature, rendering the model incapable of differentiating between the proteins with the same  $T_{og}$ . The best the model can achieve is to learn the mean value of  $T_m$  within all proteins with the same  $T_{og}$ . However, the figure shows that OGT only learned the mean value for proteins with lower  $T_{og}$ .  $T_m$  for proteins with higher  $T_{og}$  are underestimated.

Figure 17 and Figure 18 show the predicted  $T_m$  of DYN against true values on the training and validation datasets, respectively. The species of the host is color-coded in both figures. It can be seen in both figures that the distribution of

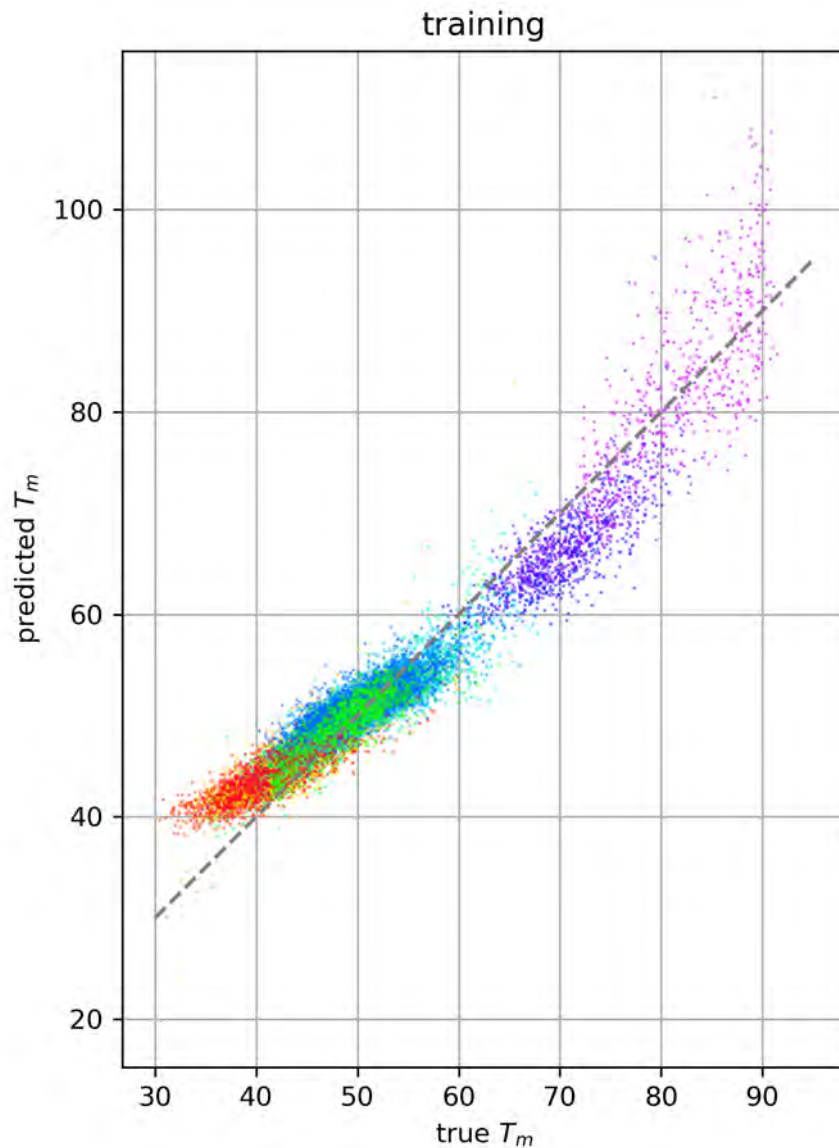




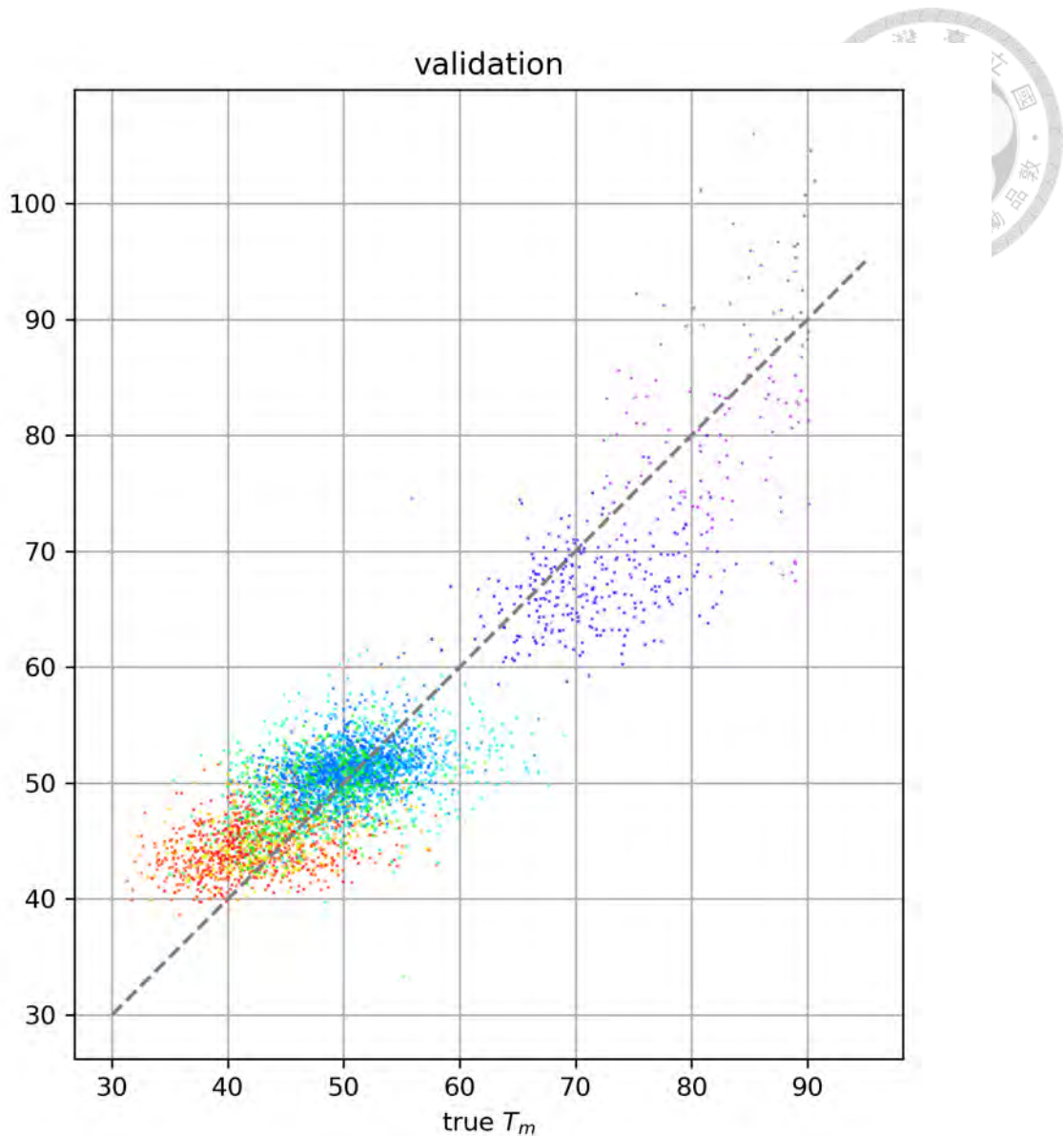
proteins in the same species does not follow the dashed line ( $T_m = T_{og}$ ). The model's bias is generally high for proteins with lower  $T_m$  but gets better for proteins with higher  $T_m$ . On the other hand, the variance of the model decreases as true  $T_m$  increases. The high bias for proteins with lower  $T_m$  suggests that the GNN architecture designed is unable to learn the underlying relation in this region. The higher variance for proteins with high  $T_m$  is possibly due to the lower number of data entries in this part of the dataset compared to proteins with lower  $T_m$  (see Figure 6). Furthermore, the direction of bias is different for proteins with low  $T_m$  and those with mid-range  $T_m$ . DYN tends to overestimate  $T_m$  for low thermostability proteins, while it tends to overestimate  $T_m$  for proteins with  $T_m$  around 60°C. The bias on the training and validation sets seem similar in the two figures, reflecting the similar data distribution in both datasets.



**Figure 16.** True  $T_m$  versus predicted  $T_m$  of the OGT model on the training and validation dataset. Stratification is inevitable due to  $T_{og}$  being the only input feature. (Each color corresponds to one of the 12 species in the dataset.)




**Figure 17.** True  $T_m$  versus predicted  $T_m$  of the DYN model on the training set. Within proteins of the same  $T_{og}$  (some species share the same  $T_{og}$ ), deviation from the true value increases as the true  $T_m$  moves away from the mean  $T_m$ . This is especially prominent for the species with lower  $T_{og}$ . Variance is higher for proteins with higher true  $T_m$ , which might result from the sparse training data in this region. (Each color corresponds to one of the 12 species in the dataset.)



**Figure 18.** True  $T_m$  versus predicted  $T_m$  of the DYN model on the validation set. The distribution is similar to that on the training set in Figure 17.

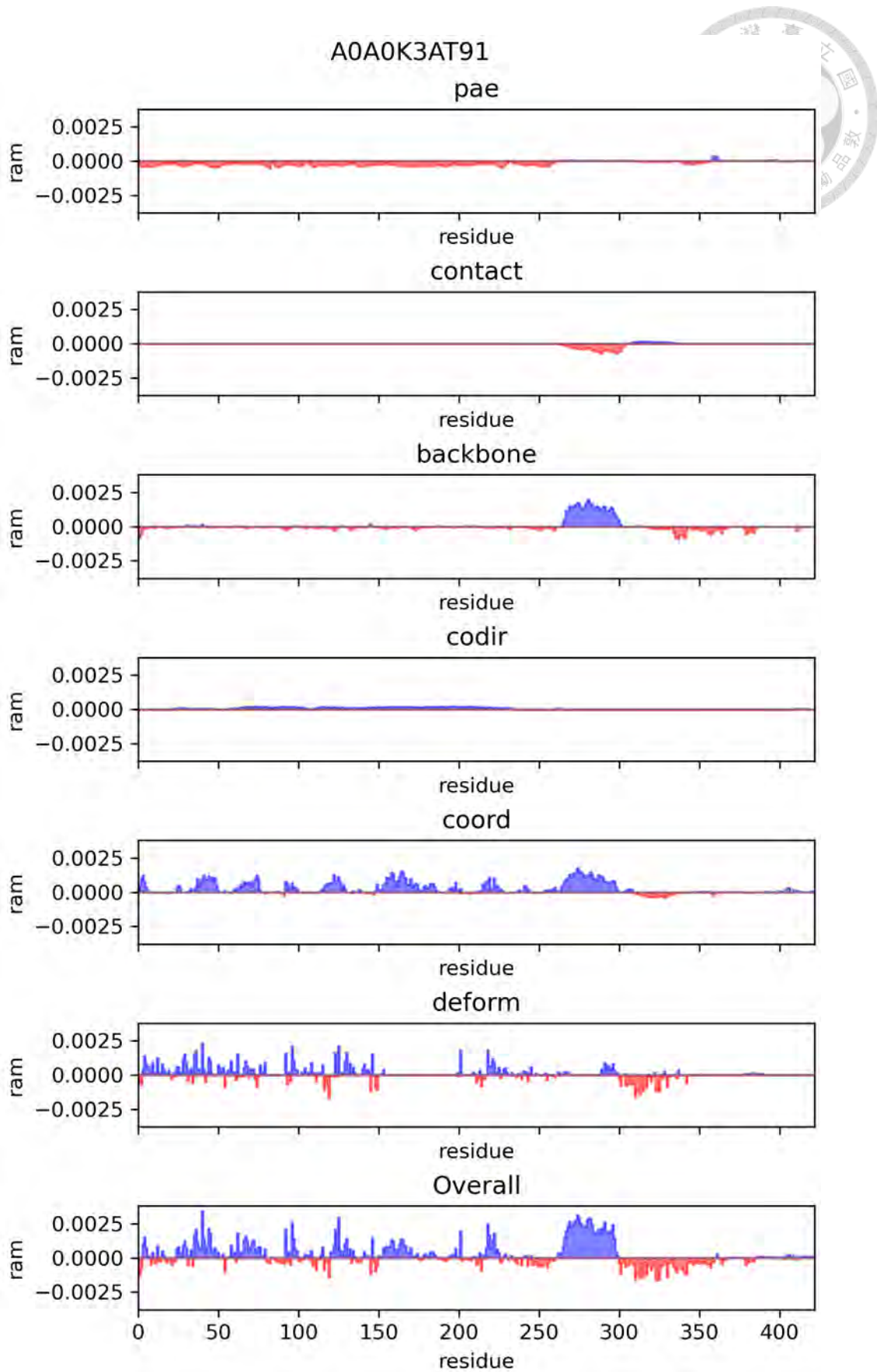
### 3.3 Structural and Dynamical Influence on Protein Thermostability

Gradient-weighted regression activation maps (Grad-RAM), as defined in 2.5 Regression Activation Map (RAM), are applied to analyze each residue's



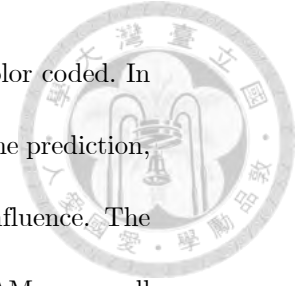
importance to the predicted value. Herein lies one of the most significant advantages of having multiple properties in the data representation: apart from providing more information to the regression model, the importance of the different features can be separated thanks to the distinct pathways each feature component provides.

An example of the influence of individual residues in a protein is shown in Figure 19 (accession number: A0A0K3AT91). In this sample, the predicted  $T_m$  is the least dependent on the co-directionality graph and the most dependent on the coordination and deformation graphs. Positive RAM on a residue implies that the residue positively influences the prediction and vice versa. Groups of RAM with the same sign positioned close to each other suggest a region on the protein that is either locally stable or unstable, depending on the sign of the values. As such, regions with greater influence on thermostability can be interpreted, indicating a possible region of interest for downstream research, such as conducting molecular dynamic simulations. The overall importance of each residue is obtained by summing the RAM values across all graphs, as shown in the bottom-most subplot. The values indicate critical locations that should be the focus of modification when tuning the thermostability of a protein.

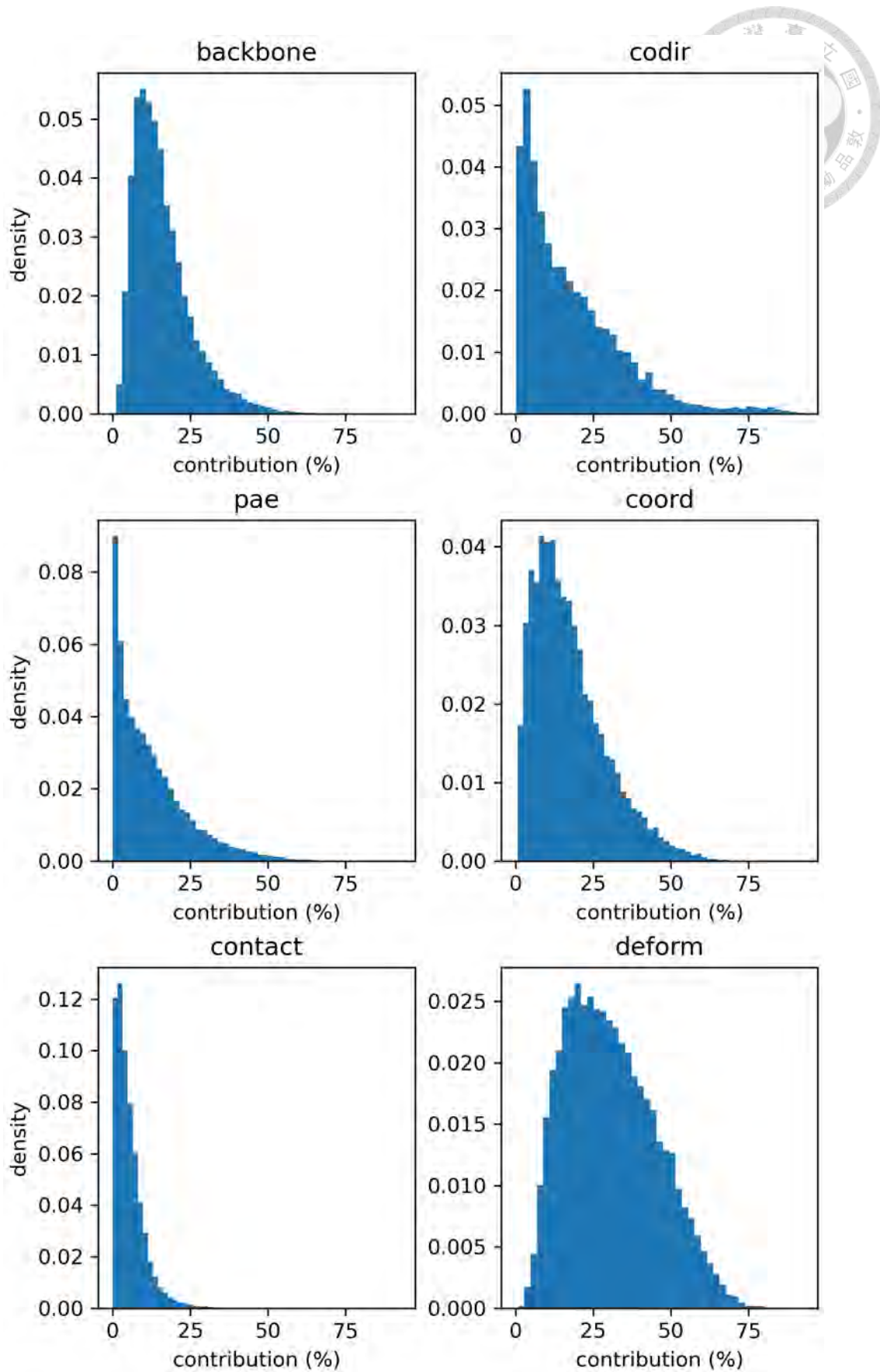


**Figure 19.** RAM for the six types of graphs for a protein (accession:

A0A0K3AT91), where the positive and negative influences are color coded. In this case, co-directionality graphs have the smallest influence on the prediction, while coordination and deformation graphs have the largest influence. The overall influence of each residue can be obtained by summing RAM across all six graphs, as shown in the lowermost row (Overall).



For each protein, the importance of each of the graphs can be computed as a percentage by summing the values across all residues. Statistics of the importance of each aspect are shown as a histogram in [Figure 20](#) and plotted against  $T_m$  in [Figure 21](#). It can be seen here that the model relies more heavily on the dynamical graphs for most proteins in the dataset and is especially reliant on the deformation graphs. However, overfitting of the model as more features are included also implies that this might be a source of noise.

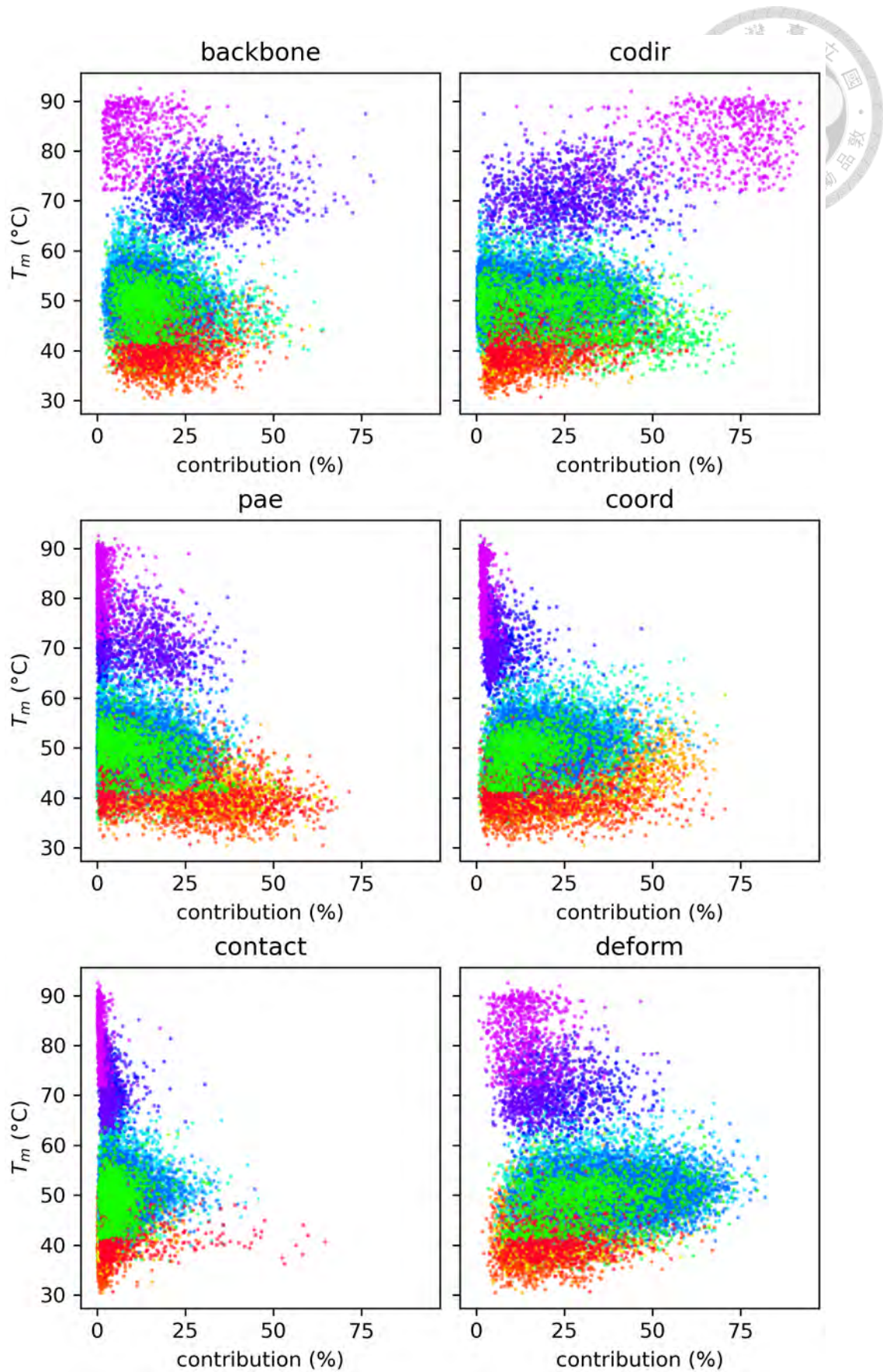


**Figure 20.** Contribution of the six different graphs to the prediction of  $T_m$  for



each protein. Overall, the model is most reliant on deformation graphs and least reliant on contact graphs.





**Figure 21.** Contribution of each type of graph to the prediction versus  $T_m$ . The

contribution of co-directionality graphs on the prediction is out of proportion

for high  $T_m$  proteins.



The implications of RAM are inherited from how the graphs are defined and computed. However, how exactly the highlighted residues based on each type of graph differ is unclear. While it can be deduced that the residues highlighted on the contact graph are indeed more relevant to protein structure than co-directionality graphs, how such a residue correlates more to structures than dynamics remains unanswered. This, however, is strongly coupled with the open question of how data are interpreted in the black box known as “machine learning.”

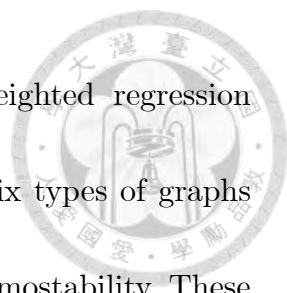
In summary, sequential, structural, and dynamical contributions to the thermostability of a protein can be analyzed with the use of the data representation defined. RAM highlights regions highly influential to thermostability, making it a practical indicator for the regions that require attention in further research. This might include simulations to uncover details in the behavior of the protein or modification to enhance the protein’s thermostability.

# Chapter 4.



## Conclusion and Outlook

This work incorporates protein dynamics into machine learning for the prediction of protein thermostability, an attribute known to be heavily influenced by the biomolecule's dynamical behavior, for the first time. A graph representation of proteins is proposed to succinctly encode protein dynamics alongside its sequential and structural features. The use of AlphaFold structures, as opposed to experimentally solved protein structures, enables the method to operate on any protein sequences. A graph neural network (GNN) model is designed to process these graph representations and make a prediction of the melting temperature  $T_m$  of the protein, achieving an MAE of 3.291°C, RMSE of 4.286°C, and  $R^2$  of 0.805 on the validation dataset. Comparison with the state-of-the-art method in the literature shows that the proposed method is equally performant. The results confirm that the representation proposed is effective in facilitating graph learning, despite the model suffering slightly from the overfitting issue notorious of GNNs. A 28-fold reduction in tunable parameters compared to the state-of-the-art model further highlights the superior efficiency in the



combination of graph representation and GNN. Gradient-weighted regression activation maps (Grad-RAM) computed individually for the six types of graphs reveal the importance of different protein attributes to its thermostability. These values provide insight into potential residues for modification that can lead to better thermostability. Statistics of RAM across the entire dataset point out that the model relies heavily on dynamical graphs, again suggesting its importance in predicting protein thermostability.

## Future Work

The graph convolution in this work is agnostic of the edges of other graphs. The model's capacity could be enhanced if the GCN layers are replaced by aggregation methods that make use of connections made in different graphs. However, this usually comes at the expense of a significant increase in tunable parameters and a higher risk of overfitting. The tradeoff will have to be mitigated.

The data representation could be revised with the help of RAM. Analysis of RAM across all proteins in the dataset could clarify which graphs are helpful for thermostability and which are not. This could serve as an indicator to find a subset of proteins that the model excels on. Research to uncover why can be conducted accordingly, and the results would influence the selection of protein

features in the future.

Graph representations are expected to perform better than sequential embeddings in modeling the effect of point mutations because of the additional connections between residues. These connections provide pathways to amplify the impact of a point mutation, which is local in the sequence but could have far-reaching effects on multiple residues within the protein. Therefore, the representation proposed in this work is suitable for predicting the change in thermostability upon mutation.

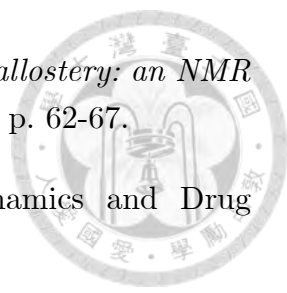
Lastly, the data representation could be used in other cases of protein property prediction. A combination of function prediction and melting temperature prediction could lead to novel biocatalyst design methods. Further research is in order.



# Bibliography

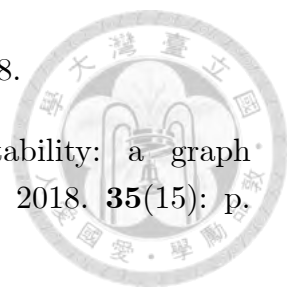


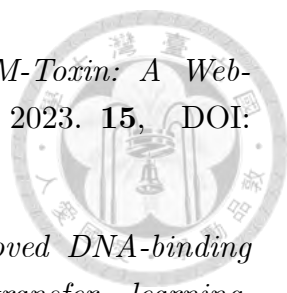
- [1] Chapman, J., A.E. Ismail, and C.Z. Dinu, *Industrial Applications of Enzymes: Recent Advances, Techniques, and Outlooks*. Catalysts, 2018. **8**(6): p. 238.
- [2] Bommarius, A.S. and M.F. Paye, *Stabilizing biocatalysts*. Chemical Society Reviews, 2013. **42**(15): p. 6534-6565.
- [3] Xu, Z., et al., Recent advances in the improvement of enzyme thermostability by structure modification. Critical Reviews in Biotechnology, 2020. **40**(1): p. 83-98.
- [4] Nezhad, N.G., et al., *Thermostability engineering of industrial enzymes through structure modification*. Applied Microbiology and Biotechnology, 2022. **106**(13): p. 4845-4866.
- [5] Ahmed, Z., H. Zulfiqar, L. Tang, and H. Lin, *A Statistical Analysis of the Sequence and Structure of Thermophilic and Non-Thermophilic Proteins*. International Journal of Molecular Sciences, 2022. **23**(17): p. 10116.
- [6] Polizzi, K.M., A.S. Bommarius, J.M. Broering, and J.F. Chaparro-Riggers, *Stability of biocatalysts*. Current Opinion in Chemical Biology, 2007. **11**(2): p. 220-225.
- [7] Pucci, F., J.M. Kwasigroch, and M. Rooman, SCooP: an accurate and fast predictor of protein stability curves as a function of temperature. Bioinformatics, 2017. **33**(21): p. 3415-3422.
- [8] Pucci, F., M. Dhanani, Y. Dehouck, and M. Rooman, Protein Thermostability Prediction within Homologous Families Using Temperature-Dependent Statistical Potentials. PLOS ONE, 2014. **9**(3): p. e91659.
- [9] Roberts, G., The role of protein dynamics in allosteric effects—introduction. Biophysical Reviews, 2015. **7**(2): p. 161-163.
- [10] Loutchko, D. and H. Flechsig, Allosteric communication in molecular machines via information exchange: what can be learned from dynamical modeling. Biophysical Reviews, 2020. **12**(2): p. 443-452.

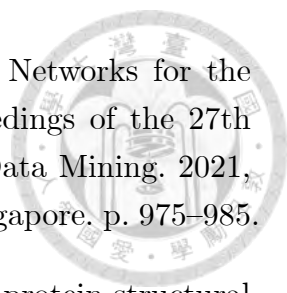
- 
- [11] Tzeng, S.-R. and C.G. Kalodimos, *Protein dynamics and allostery: an NMR view*. Current Opinion in Structural Biology, 2011. **21**(1): p. 62-67.
- [12] Peng, J.W., Communication Breakdown: Protein Dynamics and Drug Design. Structure, 2009. **17**(3): p. 319-320.
- [13] Mittag, T., L.E. Kay, and J.D. Forman-Kay, *Protein dynamics and conformational disorder in molecular recognition*. Journal of Molecular Recognition, 2010. **23**(2): p. 105-116.
- [14] Berendsen, H.J.C. and S. Hayward, *Collective protein dynamics in relation to function*. Current Opinion in Structural Biology, 2000. **10**(2): p. 165-169.
- [15] Kumar, S., D. Seth, and P.A. Deshpande, *Molecular dynamics simulations identify the regions of compromised thermostability in SazCA*. Proteins: Structure, Function, and Bioinformatics, 2021. **89**(4): p. 375-388.
- [16] Karshikoff, A., L. Nilsson, and R. Ladenstein, *Rigidity versus flexibility: the dilemma of understanding protein thermal stability*. The FEBS Journal, 2015. **282**(20): p. 3899-3917.
- [17] Tang, Q.-Y. and K. Kaneko, Long-range correlation in protein dynamics: Confirmation by structural data and normal mode analysis. PLOS Computational Biology, 2020. **16**(2): p. e1007670.
- [18] Schlick, T. and S. Portillo-Ledesma, *Biomolecular modeling thrives in the age of technology*. Nature Computational Science, 2021. **1**(5): p. 321-331.
- [19] Ku, T., et al., *Predicting melting temperature directly from protein sequences*. Computational Biology and Chemistry, 2009. **33**(6): p. 445-450.
- [20] Gorania, M., H. Seker, and P.I. Haris. Predicting a protein's melting temperature from its amino acid sequence. in 2010 Annual International Conference of the IEEE Engineering in Medicine and Biology. 2010.
- [21] Pucci, F. and M. Rooman, *Towards an accurate prediction of the thermal stability of homologous proteins*. Journal of Biomolecular Structure and Dynamics, 2016. **34**(5): p. 1132-1142.
- [22] Dehouck, Y., B. Folch, and M. Rooman, *Revisiting the correlation between proteins' thermoresistance and organisms' thermophilicity*. Protein

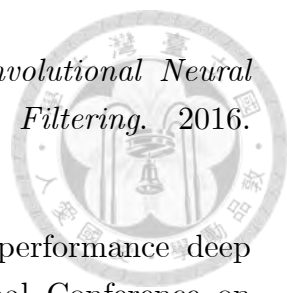


Engineering, Design and Selection, 2008. **21**(4): p. 275-278.

- 
- [23] Miotto, M., et al., Insights on protein thermal stability: a graph representation of molecular interactions. *Bioinformatics*, 2018. **35**(15): p. 2569-2577.
- [24] Jarzab, A., et al., *Meltome atlas—thermal proteome stability across the tree of life*. *Nature Methods*, 2020. **17**(5): p. 495-503.
- [25] Nikam, R., et al., ProThermDB: thermodynamic database for proteins and mutants revisited after 15 years. *Nucleic Acids Research*, 2021. **49**(D1): p. D420-D424.
- [26] Yang, Y., et al., *ProTstab – predictor for cellular protein stability*. *BMC Genomics*, 2019. **20**(1): p. 804.
- [27] Yang, Y., J. Zhao, L. Zeng, and M. Vihinen, *ProTstab2 for Prediction of Protein Thermal Stabilities*. *International Journal of Molecular Sciences*, 2022. **23**(18): p. 10798.
- [28] Jung, F., K. Frey, D. Zimmer, and T. Mühlhaus *DeepSTABp: A Deep Learning Approach for the Prediction of Thermal Protein Stability*. *International Journal of Molecular Sciences*, 2023. **24**, DOI: 10.3390/ijms24087444.
- [29] Vaswani, A., et al. *Attention Is All You Need*. 2017. arXiv:1706.03762 DOI: 10.48550/arXiv.1706.03762.
- [30] Rao, R., et al. *Evaluating Protein Transfer Learning with TAPE*. 2019. arXiv:1906.08230 DOI: 10.48550/arXiv.1906.08230.
- [31] Elnaggar, A., et al., *ProtTrans: Toward Understanding the Language of Life Through Self-Supervised Learning*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. **44**(10): p. 7112-7127.
- [32] Alley, E.C., et al., Unified rational protein engineering with sequence-based deep representation learning. *Nature Methods*, 2019. **16**(12): p. 1315-1322.
- [33] Brandes, N., et al., ProteinBERT: a universal deep-learning model of protein sequence and function. *Bioinformatics*, 2022. **38**(8): p. 2102-2110.

- 
- [34] Morozov, V., C.H.M. Rodrigues, and D.B. Ascher *CSM-Toxin: A Web-Server for Predicting Protein Toxicity*. *Pharmaceutics*, 2023. **15**, DOI: 10.3390/pharmaceutics15020431.
- [35] Aizenshtein-Gazit, S. and Y. Orenstein, *DeepZF: improved DNA-binding prediction of C2H2-zinc-finger proteins by deep transfer learning*. *Bioinformatics*, 2022. **38**(Supplement\_2): p. ii62-ii67.
- [36] Marco, A., A.P.M.d.S. Vitor, and S. Edoardo, PortPred: exploiting deep learning embeddings of amino acid sequences for the identification of transporter proteins and their substrates. *bioRxiv*, 2023: p. 2023.01.26.525714.
- [37] Senior, A.W., et al., Protein structure prediction using multiple deep neural networks in the 13th Critical Assessment of Protein Structure Prediction (CASP13). *Proteins: Structure, Function, and Bioinformatics*, 2019. **87**(12): p. 1141-1148.
- [38] Jumper, J., et al., *Highly accurate protein structure prediction with AlphaFold*. *Nature*, 2021. **596**(7873): p. 583-589.
- [39] Varadi, M., et al., AlphaFold Protein Structure Database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic Acids Research*, 2022. **50**(D1): p. D439-D444.
- [40] Zhou, J., et al., Graph neural networks: A review of methods and applications. *AI Open*, 2020. **1**: p. 57-81.
- [41] Zhang, X.-M., L. Liang, L. Liu, and M.-J. Tang, *Graph Neural Networks and Their Current Applications in Bioinformatics*. *Frontiers in Genetics*, 2021. **12**.
- [42] Husic, B.E., et al., *Coarse graining molecular dynamics with graph neural networks*. *The Journal of Chemical Physics*, 2020. **153**(19).
- [43] Strokach, A., et al., *Fast and Flexible Protein Design Using Deep Graph Neural Networks*. *Cell Systems*, 2020. **11**(4): p. 402-411.e4.
- [44] Wang, X., S.T. Flannery, and D. Kihara, *Protein Docking Model Evaluation by Graph Neural Networks*. *Frontiers in Molecular Biosciences*, 2021. **8**.

- 
- [45] Li, S., et al., Structure-aware Interactive Graph Neural Networks for the Prediction of Protein-Ligand Binding Affinity, in Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. 2021, Association for Computing Machinery: Virtual Event, Singapore. p. 975–985.
- [46] Xia, Y., C.-Q. Xia, X. Pan, and H.-B. Shen, GraphBind: protein structural context embedded rules learned by hierarchical graph neural networks for recognizing nucleic-acid-binding residues. *Nucleic Acids Research*, 2021. **49**(9): p. e51-e51.
- [47] Réau, M., N. Renaud, L.C. Xue, and A.M.J.J. Bonvin, DeepRank-GNN: a graph neural network framework to learn patterns in protein–protein interfaces. *Bioinformatics*, 2022. **39**(1).
- [48] Gligorijević, V., et al., Structure-based protein function prediction using graph convolutional networks. *Nature Communications*, 2021. **12**(1): p. 3168.
- [49] Chiang, Y., W.-H. Hui, and S.-W. Chang, *Encoding protein dynamic information in graph representation for functional residue identification*. *Cell Reports Physical Science*, 2022. **3**(7): p. 100975.
- [50] Mendez, R. and U. Bastolla, Torsional Network Model: Normal Modes in Torsion Angle Space Better Correlate with Conformation Changes in Proteins. *Physical Review Letters*, 2010. **104**(22): p. 228103.
- [51] Eckart, C., Some Studies Concerning Rotating Axes and Polyatomic Molecules. *Physical Review*, 1935. **47**(7): p. 552-558.
- [52] Murray-Rust, P. *Peptide Torsion Angles and Secondary Structure*. 1996 [cited 2023 July 20]; Available from: [https://www.cryst.bbk.ac.uk/PPS95/course/9\\_quaternary/3\\_geometry/torsion.html](https://www.cryst.bbk.ac.uk/PPS95/course/9_quaternary/3_geometry/torsion.html).
- [53] Alfayate, A., C. Rodriguez Caceres, H. Gomes Dos Santos, and U. Bastolla, *Predicted dynamical couplings of protein residues characterize catalysis, transport and allostery*. *Bioinformatics*, 2019. **35**(23): p. 4971-4978.
- [54] Kipf, T.N. and M. Welling *Semi-Supervised Classification with Graph Convolutional Networks*. 2016. arXiv:1609.02907 DOI: 10.48550/arXiv.1609.02907.

- 
- [55] Defferrard, M., X. Bresson, and P. Vandergheynst *Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering.* 2016. arXiv:1606.09375 DOI: 10.48550/arXiv.1606.09375.
- [56] Paszke, A., et al., PyTorch: an imperative style, high-performance deep learning library, in Proceedings of the 33rd International Conference on Neural Information Processing Systems. 2019, Curran Associates Inc. p. Article 721.
- [57] Fey, M. and J.E. Lenssen *Fast Graph Representation Learning with PyTorch Geometric.* 2019. arXiv:1903.02428 DOI: 10.48550/arXiv.1903.02428.
- [58] Selvaraju, R.R., et al. *Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization.* 2016. arXiv:1610.02391 DOI: 10.48550/arXiv.1610.02391.