UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

MASTER'S DEGREE IN

ICT for Internet and Multimedia

Valorizing Food Ingredients Using Machine Learning Models

Supervisor
Prof. Michele Rossi

Supervisor
Prof. Mikkel N. Schmidt

Candidate
Mohammad Vanaei

Academic Year 2024-2025

Acknowledgements

I would like to express my heartfelt gratitude to my supervisors, Professor Mikkel N. Schmidt from DTU, Dr. Søren Drud-Heydary from Arla Foods Ingredients, and Professor Michelle Rossi from Unipd for their invaluable support, guidance, and encouragement throughout my research.

I am also deeply grateful to my dearest family and friends for their unwavering support and for always being there for me.

Abstract

The classification of peptides, particularly antibacterial peptides, is a critical task in bioinformatics with applications in drug discovery and therapeutic development. This study explores the use of Graph Neural Networks (GNNs), specifically Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs), to classify peptides based on their functional properties. Unlike traditional sequence-based approaches, we incorporate structural information by constructing peptide graphs from 3D structures generated using AlphaFold.

We evaluate multiple feature representations, including one-hot encoding and physicochemical properties, to assess their impact on model performance. Experimental results indicate that GCN and GAT models can effectively classify peptides, with GCN achieving higher accuracy but exhibiting overfitting tendencies. The integration of additional features contributed to model stability but did not lead to substantial performance gains. Our findings suggest that combining structural insights with deep learning can enhance peptide classification, but limitations such as the availability of experimentally validated 3D structures and sequence redundancy remain challenges.

Future work should focus on refining graph representations, incorporating advanced transformer-based models, and improving computational 3D structure predictions. Enhancing dataset diversity and leveraging hybrid architectures may further improve classification accuracy and generalization.

Keywords: Peptide, Machine Learning, Graph Neural Network, GCN, Antibacterial

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Peptides

Peptides are short chains of amino acids linked together by peptide bonds and they are produced by different types of cells within most organisms (etc. bacteria, plants, humans). Peptides play various roles in biological systems, serving as signaling molecules, hormones, neurotransmitters, and components of larger proteins. They provide essential amino acids necessary for growth, repair and maintenance of body tissues. They are essential for numerous physiological processes in living organisms, including muscle contraction, immune response, and cell signaling. They also have some functional properties like foaming (peptides aid in the formation and stability of foams) or emulsification (peptides can stabilize emulsions in products like salad dressings and sauces) and also flavour enhancement.

1.1.1 Bioactive (or Functional) Peptides

Bioactive peptides are short amino acid sequences derived from proteins that exhibit specific biological activities and physiological effects when consumed. These activities may include antioxidant, antimicrobial, antihypertensive, anti-

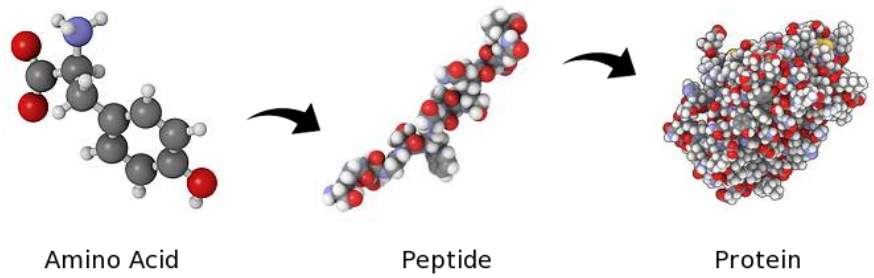


Figure 1.1: Proteins and peptides formation
(https://www.peptidesciences.com/peptide-information/peptides-vs-proteins)

inflammatory, immunomodulatory, opioid, calcium absorption and bone health, anticancer or wound healing.[1]
Bioactive peptides are often released from larger proteins through enzymatic digestion or food processing techniques. E.g. digestive enzymes in the stomach and intestine will release peptides from the proteins in the food we consume – these peptides that are released in our gastrointestinal tract could very likely also have a bioactive potential. They can be found naturally in various foods, such as milk, eggs, fish, and grains. They also can be produced through enzymatic or chemical synthesis for use in functional foods, dietary supplements, or pharmaceuticals.
Studying peptides can help us to understand and be able to explain why different proteins have different functionalities and outcomes.

Therapeutically, there are many benefits from bioactive peptides that make them more useful than traditional medicines based on small molecules. Bioactive peptides have more specialized activities on the target tissue and therefore have little or no toxic effects which is especially important in treating chronic diseases. One of the most interesting and considerable benefits of antimicrobial peptides is that they are less prone to causing antibiotic resistance which is becoming a major issue. Unlike antibiotics, AMPs do not readily cause resistance. Also, the development of new antibiotics has become more challenging, so AMPs can be a good alternative.[1]

Potential health benefits and also therapeutic properties of peptides have made them popular, and much research have been trying to understand what the potential of peptides are, what the most important features of peptides are, how these features can lead to a different functionality and of course how they can be used to have desired outcome.

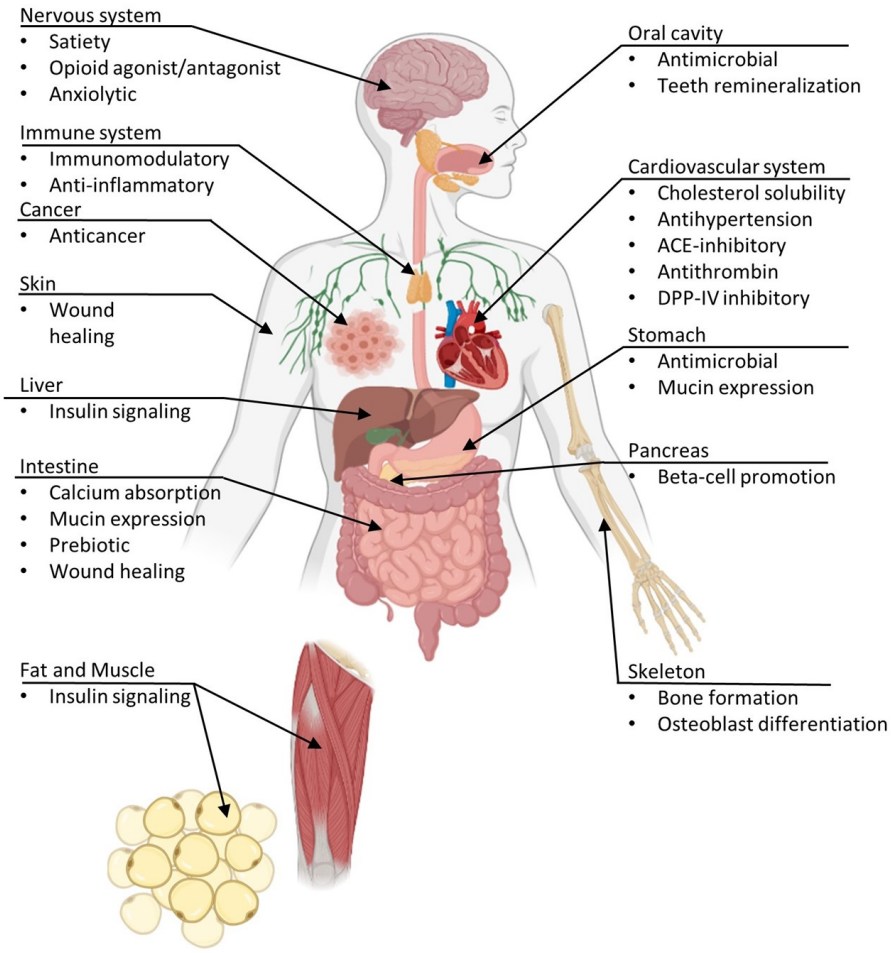


Figure 1.2: Illustration of the proposed sites of action for milk-derived peptides, highlighting their specific biological activities in various systems of the human body [2].

1.2 Amino Acids: The Building Blocks

Amino acids areany of a group of organic molecules that consist of a basic amino group (—NH2), an acidic carboxyl group (—COOH), and an organic R group (or side chain) that is unique to each amino acid. They are the fundamental units that build peptides and proteins. They contain both amino and carboxylic

acid functional groups. There are two types of amino acids: essential (which can not be produced by our body) and non essential (which out body can produce).

There are 20 standard amino acids, each with unique properties and functions, forming the diverse sequences that determine the specific structure and function of the resulting peptides and proteins. These amino acids include:

One-letter Code	Three-letter Code	Full Name
A	Ala	Alanine
R	Arg	Arginine
N	Asn	Asparagine
D	Asp	Aspartic Acid
C	Cys	Cysteine
E	Glu	Glutamic Acid
Q	Gln	Glutamine
G	Gly	Glycine
H	His	Histidine
I	Ile	Isoleucine
L	Leu	Leucine
K	Lys	Lysine
M	Met	Methionine
F	Phe	Phenylalanine
P	Pro	Proline
S	Ser	Serine
T	Thr	Threonine
W	Trp	Tryptophan
Y	Tyr	Tyrosine
V	Val	Valine

Table 1.1: 20 Standard amino acids

Each molecule has a central Carbon (C), called the α-carbon, to which both an amino and a carboxyl group are attached. WE will use this central carbon can be used for geometric analysis.

1.3 Machine Learning Models for Predicting Peptides Functionalities

As highlighted, peptides play a pivotal role in bioactivity and offer potential in various medications with minimal side effects. This motivates us to delve

into understanding the diverse types of peptides, their unique functionalities, and characteristics, such as taste, to facilitate the synthesis of amino acids for creating effective bioactive peptides to have nutritious products. There are several ways to do this:

- Lookup in databases
- Using some complex specific protein modelings
- Prediction

The problem is the fact that amino acids can vary in length from 2 to 100 and there are 20 standard amino acids, which is a very huge number of different variations. So it is not feasible to test all of these variations and have a database (which is also needs to e validated). Moreover, the significant expenses associated with extracting and manufacturing peptides underscore the necessity for an alternative approach to predict their functionalities. Thorough investigation and precise prediction of peptide functional classes hold the potential to substantially reduce manufacturing costs and furnish invaluable insights for the development of novel drugs. Prediction models have been developed to screen large peptide datasets for potential bioactivity. These models rely on data derived from laboratory studies confirming the biological activity of peptides. A crucial aspect of constructing these prediction models involves assessing how various descriptive factors associated with each amino acid, positioned within the peptide sequence, correlate with their bioactive functions.

Numerous studies have tried to forecast the functional classes of peptides. They differ in their methodology, models used, feature selection, single or multi label classification and also some other variation. The advancement of machine learning, particularly deep learning models has showed the possibility of low cost and more accurate prediction.

This thesis focuses on leveraging these advanced computational techniques to classify and estimate peptide functionality, particularly their antimicrobial properties. This thesis investigates the application of deep learning models to classify and predict peptide functionality, specifically focusing on their antibacterial properties. The objective is to develop a model informed by prior research on peptides and molecules, aiming to identify the most critical features that influence peptide behavior and achieve more accurate prediction outcomes.

1.4 Thesis Structure

This thesis is organized as follows:

Chapter 1: Introduction Overview of peptides and proteins and their importance and also motivation of this study

Chapter 2: Literature Review Overview of machine learning in biological classification; machine learning approaches for peptide classification; evaluation methods.

Chapter 3: Methodology Data, graph representation of peptide data; Graph Neural Networks (GNNs) and special models have been used; methods for constructing peptide graphs out of the geometry, features of the data

Chapter 4: Results and Discussion Model architectures (GCN and GAT); hyperparameter tuning; performance evaluation metrics; analysis of results.

Chapter 2

Literature Review

Machine learning (ML) is a powerful tool for predicting biological functionalities and properties. ML models can learn from existing data and make accurate predictions about peptides.

ML tools like support vector machines (SVMs), random forests, and neural networks work well with complex biological data. These tools excel at handling diverse types of information simultaneously. An important part of using ML is selecting which pieces of data are most useful, a process known as feature selection.

There is a lot of research in the area of peptide functionality prediction, which is growing due to the importance of peptides. Some of this research has led to the development of freely available tools for prediction, such as AMPER[4], iAMPpred[5](examples of conventional machine learning), and AMPlify[12], APIN[13] (examples of deep learning approaches).

These approaches show high diversity in their dataset size, data quality, core algorithms, feature extraction, feature selection techniques, and evaluation strategies.[3]

Several ways exist to categorize these studies:

1. Based on the models used.

2. Based on the features used as input to the models, which is a key factor in the models' accuracy.
3. Based on whether they perform single-label or multi-label classification.
4. Based on the datasets used, which can also lead to very different results.

In general, we can divide these machine learning studies into two groups: conventional machine learning and deep learning models.

2.1 Overview of Machine Learning in Biological Classification

Machine learning (ML) techniques have transformed biological classification by providing powerful analytical tools for complex biological data. ML enables the creation of predictive models that learn from existing data to make accurate predictions on new, unseen data. Across various biological classification tasks, such as identifying genes, proteins, peptides, predicting protein structure and function, and classifying diseases, ML has found widespread applications[3, 1]

Algorithms like support vector machines (SVM), random forests (RF), and neural networks have proven particularly effective in handling the high-dimensional and heterogeneous nature of biological data. Feature engineering and selection play crucial roles in ML workflows, allowing the extraction of relevant information from biological sequences. Moreover, the advent of deep learning has further enhanced the accuracy and applicability of ML models in biology by enabling automatic learning of complex feature representations from raw data [3, 1]

2.2 Machine Learning Models in Peptide Classification

Peptide classification, particularly the identification and functionality prediction of antimicrobial peptides (AMPs), has significantly benefited from advancements in machine learning techniques. The complexity and diversity of peptides necessitate sophisticated computational approaches to complement traditional experimental methods, which are often labor-intensive and time-consuming. This section reviews the key machine learning methodologies applied to peptide

classification, emphasizing their evolution, implementation, and performance [3, 1, 14].

2.2.1 Conventional Machine Learning Approaches

Early efforts in peptide classification utilized conventional machine learning algorithms, such as support vector machines (SVM), random forests (RF), and k-nearest neighbors (k-NN). These methods rely heavily on feature engineering, where domain knowledge is used to extract relevant features from peptide sequences.

2.2.1.1 Support Vector Machines (SVM)

SVMs are popular due to their robustness in handling high-dimensional data. They have been widely used in tools like CAMP, which applies SVMs along with other algorithms (RF, ANN) for AMP prediction. The incremental feature selection (IFS) and recursive feature elimination (RFE) methods are often employed to optimize the feature set, improving the classifier's performance [3, 15].

2.2.1.2 Random Forests (RF)

RF classifiers are favored for their ability to handle large datasets with many features. They aggregate the results of multiple decision trees to enhance predictive accuracy and control over-fitting. Tools like AmPEP and dbAMP utilize RF algorithms to classify AMPs by employing a diverse range of sequence-derived and physicochemical features [1, 3, 16].

2.2.1.3 k-Nearest Neighbors (k-NN)

The k-NN algorithm, used in models like iAMP-2L, classifies peptides based on the closest training examples in the feature space. Fuzzy k-NN variants have been introduced to account for the inherent uncertainties in peptide classification, improving the model's robustness [3, 17].

2.2.2 Deep Learning Approaches

The advent of deep learning (DL) has revolutionized peptide classification by enabling the automatic extraction of complex features directly from raw sequences. DL models, particularly convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have shown remarkable success in bioinformatics applications.

2.2.2.1 Convolutional Neural Networks (CNNs)

CNNs are adept at capturing spatial hierarchies in data, making them suitable for peptide sequence analysis. Tools like Deep-AmPEP30 utilize CNN architectures to extract features from peptide sequences and classify them with high accuracy [14, 21].

2.2.2.2 Recurrent Neural Networks (RNNs)

RNNs, including long short-term memory (LSTM) networks, are designed to handle sequential data, making them ideal for modeling peptide sequences. The AMP Scanner v.2 leverages LSTMs to predict AMP functionalities by learning the dependencies between amino acids in a sequence [14, 22].

2.2.2.3 Attention Mechanisms

Recent advancements incorporate attention mechanisms into DL models, allowing the network to focus on the most relevant parts of the input sequence. AMPlify, for instance, combines Bi-LSTM and multi-head scaled dot-product attention to enhance the classification accuracy of AMPs [14, 23].

2.2.2.4 Graph Neural Networks (GNNs)

Graph Neural Networks (GNNs) have emerged as a powerful tool in peptide classification due to their ability to model complex relationships within biological data. GNNs, such as Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs), can effectively process graph-structured data, capturing intricate dependencies and interactions.

Graph Convolutional Networks (GCNs) GCNs are used to analyze graph-structured data, making them suitable for modeling the spatial relationships in peptide sequences. They have been applied in various bioinformatics applications, including protein-protein interaction prediction and drug discovery [6]. For instance, GCNs have shown success in capturing the complex structural patterns in protein-protein interaction networks, providing insights that were previously unattainable through traditional methods.

Graph Attention Networks (GATs) GATs utilize attention mechanisms to weigh the importance of different nodes and edges in a graph, enhancing the ability to capture relevant features. The sAMPpred-GAT model, for instance, uses GATs to predict AMPs by incorporating structural information from predicted peptide structures [14]. By leveraging the attention mechanism, GATs can dynamically focus on the most critical parts of the graph, improving the interpretability and performance of the predictive models.

2.3 Feature Extraction and Categorization

The features used in peptide prediction models can be categorized into five main types: **composition-based features**, **position-based features**, **structural features**, **physicochemical properties**, and **similarity-based features**.

2.3.1 Composition-Based Features

Composition-based features focus on the amino acid sequence without considering position or order information. These features include:

- **Amino Acid Composition (AAC)**: The frequency of each amino acid in a peptide sequence [5].
- **Dipeptide Composition (DPC) and Tripeptide Composition (TPC)**: Capture two- and three-residue sequence patterns, providing more context than AAC [8].
- **Normalized Amino Acid Composition (NAAC)**: A normalized version of AAC, accounting for sequence length variations [5].
- **N-gram Composition (NCC, NTC)**: Counts of overlapping subsequences (N-grams) found in peptide sequences [7].

2.3.2 Position-Based Features

Position-based features consider the sequential arrangement of amino acids in a peptide sequence. These include:

- **N-gram Binary Profiling (NCB, NTB)**: Binary representations of N-gram patterns in peptides [7].
- **Motifs Binary Profiling (MB)**: Identifies conserved sequence motifs present in AMPs [7].
- **Position-Specific Scoring Matrices (PSSM)**: Captures evolutionary conservation by aligning sequences with known peptide families [11].

2.3.3 Structural Features

Structural features capture the secondary and tertiary structures of peptides, which are important for functionality prediction. These features include:

- **α-Helix, β-Sheet, β-Turn, and Random Coil Content**: Quantification of secondary structure elements [9].
- **Loop Formation**: Determines flexibility and conformational stability in peptides [10].
- **Predicted 3D Structures**: Computationally derived structural models aid in predicting peptide interactions [14].

2.3.4 Physicochemical Properties

Physicochemical properties define the biophysical and chemical characteristics of peptides, influencing their functionality. Important properties include:

- **Charge and Isoelectric Point (pI)**: Determines peptide solubility and interaction with cellular membranes [8].
- **Hydrophobicity and Hydrophilicity**: Essential for peptide folding, stability, and membrane permeability [8].
- **Polarity and Polarizability**: Affect peptide interactions with biomolecules [9].

2.3.5 Similarity-Based Features

Similarity-based features leverage known peptide datasets to infer properties of unknown peptides:

- **BLOSUM-50 Matrices**: Score sequence similarity based on evolutionary substitutions [8].
- **Pairwise Similarity Scores**: Uses sequence alignment algorithms (e.g., BLAST) to predict function based on known peptides [14].

Chapter 3

Methodology

3.1 Graph Representation of Data

Some data are naturally presented in graph form, such as networks of people, molecules, and social interactions. For instance, in molecular structures, each atom can be considered a node, with connections (edges) representing chemical bonds. These bonds are crucial features that determine the molecule's properties. Similarly, proteins and peptides are chains of amino acids linked by peptide bonds, where the interactions between these amino acids govern their functionality. By representing a peptide as a graph, consisting of nodes (amino acids) and edges (interactions), we can uncover the key features that dictate its behavior. This graph-based representation is particularly useful in neural networks for tasks such as classification.

3.2 Graph Neural Networks (GNNs)

In the realm of machine learning, Graph Neural Networks (GNNs) stand out as a transformative technology designed to handle graph-structured data. Unlike traditional data forms, graphs consist of nodes (vertices) and edges that capture

the relationships between nodes. This unique structure is prevalent in many real-world applications, such as social networks, biological networks, and knowledge graphs. GNNs are tailored to exploit these structures, making them essential for tasks that require understanding and leveraging the intricate interdependencies within data.

3.2.1 Understanding Graph Neural Networks

Graph Neural Networks extend the capabilities of traditional neural networks to non-Euclidean domains, specifically graphs. A graph $G = (V, E)$ consists of a set of nodes V and a set of edges E connecting these nodes. Each node $v \in V$ can be associated with a feature vector $\mathbf{x}_v$, and edges $(u, v) \in E$ can have associated features $\mathbf{e}_{uv}$. GNNs operate by iteratively updating node representations (embeddings) through a process called message passing or neighborhood aggregation. This allows nodes to incorporate information from their neighbors, thus capturing the local and global structure of the graph.

3.2.2 Importance of GNNs

The significance of GNNs lies in their ability to process relational data, which is common in various domains. Unlike traditional neural networks that handle Euclidean data like images and text, GNNs excel in scenarios where the relationships between data points are as crucial as the data points themselves. This relational approach enables GNNs to capture complex dependencies and interactions, leading to enhanced performance in tasks such as node classification, link prediction, and graph classification.

3.2.3 Benefits of GNNs

1. **Capturing Local and Global Structures:** GNNs can simultaneously capture local neighborhoods and global graph structures, providing a comprehensive understanding of the graph.

2. **Permutation Invariance:** GNNs respect the permutation invariance property of graphs, ensuring that the output does not depend on the ordering of nodes or edges.

3. **Scalability:** Various GNN architectures are designed to scale with large graphs, making them suitable for big data applications.

3.2.4 Core Aspects of GNNs

Graph Neural Networks (GNNs) are built upon several fundamental operations: message passing, aggregation, update, and readout. These operations enable GNNs to capture the complex structures and relationships within graph data. Here, we delve deeper into each of these components, providing detailed explanations and relevant formulas.

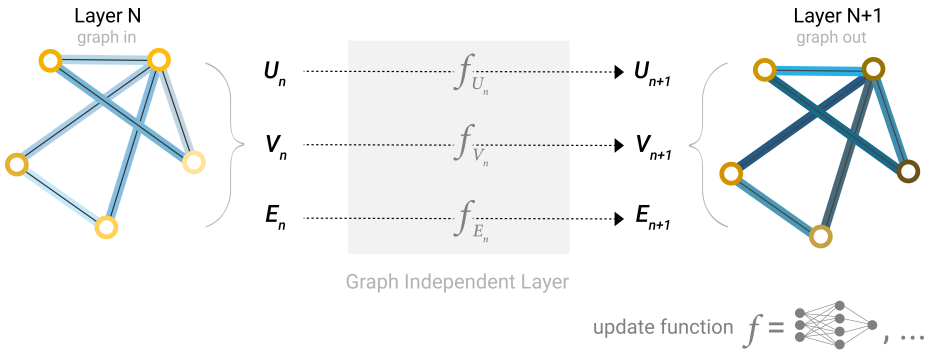


Figure 3.1: A single layer of a simple GNN.
(https://distill.pub/2021/gnn-intro/)

3.2.4.1 Message Passing

Message passing is the core mechanism in GNNs, where nodes in the graph exchange information with their neighbors to update their own representations. This process can be broken down into two main steps: message computation and message aggregation.

1. **Message Computation:**

   Each node v computes a message to send to its neighbors. The message from node u to node v at the k-th layer is typically a function of the features of node u and the edge connecting u to v:

   $$\mathbf{m}_{uv}^{(k)} = \text{Message}\left(\mathbf{h}_u^{(k-1)}, \mathbf{h}_v^{(k-1)}, \mathbf{e}_{uv}\right)$$

   where:

   - $\mathbf{h}_u^{(k-1)}$ and $\mathbf{h}_v^{(k-1)}$ are the representations of nodes u and v at layer $k-1$,

- $\mathbf{e}_{uv}$ represents the edge features between nodes u and v.

2. **Message Aggregation:**

   Each node v aggregates the messages received from its neighbors $\mathcal{N}(v)$. The aggregation function combines these messages into a single vector:

$$\mathbf{m}_v^{(k)} = \text{Aggregate}\left(\{\mathbf{m}_{uv}^{(k)} : u \in \mathcal{N}(v)\}\right)$$

   Common aggregation functions include:

   - **Sum:** $\mathbf{m}_v^{(k)} = \sum_{u \in \mathcal{N}(v)} \mathbf{m}_{uv}^{(k)}$
   - **Mean:** $\mathbf{m}_v^{(k)} = \frac{1}{|\mathcal{N}(v)|} \sum_{u \in \mathcal{N}(v)} \mathbf{m}_{uv}^{(k)}$
   - **Max:** $\mathbf{m}_v^{(k)} = \max_{u \in \mathcal{N}(v)} \mathbf{m}_{uv}^{(k)}$

3.2.4.2 Update

After aggregating the messages from its neighbors, each node v updates its representation. The update function typically involves combining the node's current representation with the aggregated message:

$$\mathbf{h}_v^{(k)} = \text{Update}\left(\mathbf{h}_v^{(k-1)}, \mathbf{m}_v^{(k)}\right)$$

A common update function is:

$$\mathbf{h}_v^{(k)} = \sigma\left(\mathbf{W}^{(k)} \cdot \left[\mathbf{h}_v^{(k-1)} \parallel \mathbf{m}_v^{(k)}\right]\right)$$

where:

- $\mathbf{W}^{(k)}$ is a learnable weight matrix,
- $\parallel$ denotes concatenation,
- σ is an activation function, such as ReLU or sigmoid.

3.2.4.3 Readout

The readout function is applied after several message-passing iterations to produce a graph-level representation. This is particularly useful for tasks like graph classification. The readout function aggregates node-level representations into a single vector:

$$\mathbf{h}_G = \text{Readout}\left(\{\mathbf{h}_v^{(K)} : v \in V\}\right)$$

Common readout functions include:

- **Sum:** $\mathbf{h}_G = \sum_{v \in V} \mathbf{h}_v^{(K)}$
- **Mean:** $\mathbf{h}_G = \frac{1}{|V|} \sum_{v \in V} \mathbf{h}_v^{(K)}$
- **Max:** $\mathbf{h}_G = \max_{v \in V} \mathbf{h}_v^{(K)}$

3.2.5 Types of GNNs

- **Graph Convolutional Networks (GCNs):** Extend the concept of convolution to graphs.
- **Graph Attention Networks (GATs):** Use attention mechanisms to weigh the importance of neighboring nodes.
- **Graph Recurrent Networks (GRNs):** Use recurrent neural networks to update node embeddings iteratively.
- **Graph Autoencoders (GAEs):** Used for unsupervised learning to encode graph data into a lower-dimensional space and decode it back.

3.2.5.1 Example: Graph Convolutional Networks (GCNs)

Graph Convolutional Networks (GCNs) are a specific type of GNN that uses a particular formulation for message passing and update steps. In a GCN, the message passing and aggregation steps are combined into a single operation:

$$\mathbf{H}^{(k+1)} = \sigma\left(\hat{A}\mathbf{H}^{(k)}\mathbf{W}^{(k)}\right)$$

where:

- $\mathbf{H}^{(k)}$ is the matrix of node features at layer k,
- $\hat{A}$ is the normalized adjacency matrix,
- $\mathbf{W}^{(k)}$ is a learnable weight matrix,
- σ is an activation function.

The normalization of the adjacency matrix $\hat{A}$ ensures that the contributions from neighboring nodes are appropriately scaled, preventing issues arising from varying node degrees.

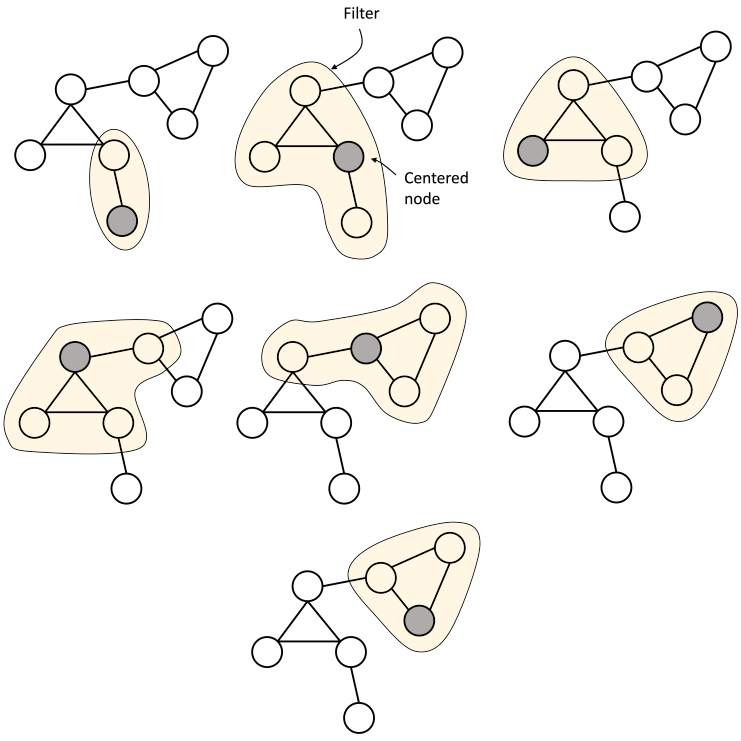


Figure 3.2: Convolution in GCN, somehow like image in CNN. (Image from here)

3.2.5.2 Example: Graph Attention Networks (GATs)

Graph Attention Networks (GATs) enhance the message passing mechanism by introducing attention mechanisms. This allows the model to weigh the importance of different neighbors dynamically. The attention coefficient α_{vu} between nodes v and u is calculated as:

$$\alpha_{vu} = \frac{\exp(\text{LeakyReLU}(\mathbf{a}^T[\mathbf{W}\mathbf{h}_v \parallel \mathbf{W}\mathbf{h}_u]))}{\sum_{k \in \mathcal{N}(v)} \exp(\text{LeakyReLU}(\mathbf{a}^T[\mathbf{W}\mathbf{h}_v \parallel \mathbf{W}\mathbf{h}_k]))}$$

where:

- $\mathbf{a}$ is a learnable weight vector,
- $\mathbf{W}$ is a learnable weight matrix,
- $\parallel$ denotes concatenation.

The node representation is then updated as:

$$\mathbf{h}_v^{(k+1)} = \sigma\left(\sum_{u \in \mathcal{N}(v)} \alpha_{vu} \mathbf{W}\mathbf{h}_u^{(k)}\right)$$

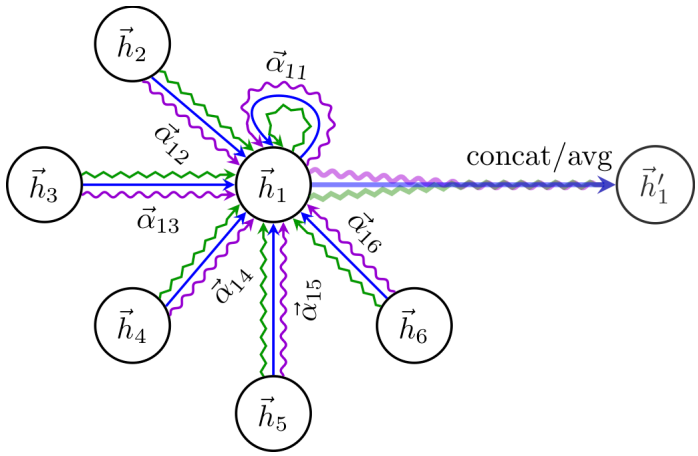


Figure 3.3: Graph Attention Networks Layer. (Image from here)

3.3 Methodology for Constructing Peptide Graphs

To effectively utilize Graph Neural Networks (GNNs) for peptide analysis, the first crucial step is constructing graphs from peptide data. Understanding the structural properties of proteins and peptides is fundamental to studying their biological activities, as their function is highly dependent on their three-dimensional shape and molecular interactions.

Each sample in our dataset is represented as a sequence of amino acids, denoted by specific letter codes. However, to leverage the full potential of GNNs, we must transform these sequences into structured molecular graphs that accurately capture spatial relationships, such as atomic distances, bond angles, and geometric conformations.

A key challenge in this process is the limited availability of experimentally validated 3D structures for peptides. Since most peptides lack high-resolution structural data from laboratory experiments, computational methods must be employed to predict their conformations with sufficient accuracy.

To address this challenge, various computational platforms and algorithms have been developed for peptide structure prediction, refinement, and validation. These tools enable the generation of realistic 3D molecular models, which serve as reliable input for downstream analysis using GNNs. In this study, we explored multiple state-of-the-art methods to computationally derive the 3D structures of peptides, ensuring that our graph-based representations accurately reflect their molecular architecture

3.3.1 Tools for Peptide Modeling

- **Rosetta:** Rosetta is a comprehensive suite of algorithms for protein modeling and computational biology. It is widely used for predicting and designing protein structures, protein folding, docking, and protein-protein interactions. In this study, Rosetta helped in refining peptide structures to ensure they are accurate and biologically relevant.

- **Pypept Library:** Pypept is a Python library designed for peptide science. It provides functionalities for peptide sequence analysis, modification, and property prediction. Using Pypept, we could preprocess peptide sequences and generate initial structural models for further refinement and analysis.

- **OmegaFold:** OmegaFold is a deep learning-based tool for predicting pro-

tein structures. It leverages advanced algorithms to generate accurate structural models from amino acid sequences. This tool was used to cross-validate the models obtained from other methods and ensure consistency in our predictions.

- **PEP-FOLD:** PEP-FOLD is a de novo peptide structure prediction server that has been widely used in the field of antimicrobial peptide research. It employs a coarse-grained approach to predict 3D structures of peptides from their amino acid sequences. Currently, the last version is PEP-FOLD4.

- **AlphaFold:** AlphaFold is an artificial intelligence system developed by DeepMind that predicts protein structures with remarkable accuracy. It has revolutionized the field of structural biology by providing high-quality predictions that often match experimental results. AlphaFold was a critical tool in our study, and we utilized it extensively for peptide modeling.

  AlphaFold employs a deep learning approach to predict the three-dimensional (3D) structure of proteins from their amino acid sequences. It uses a neural network architecture that integrates information from homologous sequences, multiple sequence alignments, and structural templates to generate accurate models.

3.3.2 Structural Validation and Evaluation of Computed 3D Structures

Once the 3D peptide structures have been computationally generated, the next critical step is to evaluate their accuracy and reliability before integrating them into our models. To achieve this, we leveraged experimentally validated 3D structures from the **RCSB Protein Data Bank (PDB)**, a widely recognized repository containing high-resolution structural data for biological macromolecules. These experimentally determined structures served as a benchmark for assessing the accuracy of our computational predictions.

To validate the predicted structures, we employed a two-step approach: visual inspection and quantitative analysis. For visual evaluation, we used molecular visualization tools (PyMOL) to compare the predicted conformations against experimentally determined structures. Additionally, we calculated the Root Mean Square Deviation (RMSD) using PyMOL, which quantifies the average atomic displacement between the predicted and reference structures. A lower RMSD indicates a higher degree of structural similarity.

Among all the computational tools tested, AlphaFold consistently produced

predictions that closely matched the validated structures, making it the most reliable tool for our study. AlphaFold generates five structural models for each peptide, from which we selected the top-ranked model based on confidence scores for further analysis. These models are provided as PDB (Protein Data Bank) files, which contain the atomic coordinates necessary for structural representation.

Finally, we extracted atomic coordinates from the PDB files to construct peptide graphs, ensuring that our models incorporate accurate spatial and physicochemical features for downstream graph-based learning.

3.3.3 Graph Construction from AlphaFold Models

To construct the peptide graphs, we utilized the coordinates from the AlphaFold-generated PDB files. Specifically, we focused on the Cα (alpha carbon) atoms for calculating distances and constructing the graph. Here is a step-by-step breakdown of the process:

1. **Distance Calculation:** For graph construction, we utilized a contact map approach based on Cα atoms distances. We calculated the Euclidean distance between the Cα atoms of each pair of residues. A threshold distance of 10 Å (Ångströms) was used to determine whether an edge should be created between two nodes (residues). If the distance between two Cα atoms was less than or equal to 10 Å, an edge was established between the corresponding nodes.
2. **Angle Calculation:** In addition to distances, we calculated three angles to capture the geometric relationships between residues. These angles were also used as edge attributes, providing additional structural information for the GNN to leverage.

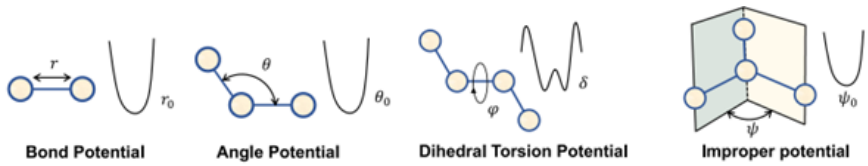


Figure 3.4: Types of Molecular Potential Energy Terms[18]

3. **Graph Representation:** Each peptide was represented as a graph where:
    - Nodes correspond to the residues (amino acids) of the peptide.

- Edges represent connections between residues based on the distance threshold and the calculated angles.
- Edge attributes included the distances and angles between residues.

By constructing graphs in this manner, we captured both the spatial and geometric properties of the peptides, enabling a detailed analysis using GNNs. This approach allows us to explore the complex relationships within peptide structures and predict their properties more accurately.

3.4 Features of Data

In this study, we utilized several features for representing the peptides and their interactions. These features included one-hot encoding, positional features, and Position-Specific Scoring Matrix (PSSM), along with physicochemical properties. Additionally, we included three angles and distance as edge features.

3.4.1 Node Features

1. **One-Hot Encoding:** One-hot encoding is used to represent the amino acid sequence of peptides. Each amino acid is represented by a vector of length 20, where the position corresponding to the amino acid is set to 1, and all other positions are set to 0.

2. **Positional Features:** Positional features capture the position of each amino acid within the peptide sequence. This is often encoded using sinusoidal functions to allow the model to differentiate between different positions.

$$\mathrm{PE}(pos, 2i) = \sin\left(\frac{pos}{10000^{2i/d}}\right)$$
$$\mathrm{PE}(pos, 2i+1) = \cos\left(\frac{pos}{10000^{2i/d}}\right)$$

   where pos is the position, i is the dimension, and d is the total dimension of the positional encoding.

3. **Position-Specific Scoring Matrix (PSSM):**

   Position-Specific Scoring Matrix (PSSM) is a powerful tool used to capture the evolutionary information of amino acid sequences. It is generated by aligning a query sequence with homologous sequences in a database

and calculating the frequency of each amino acid at each position. This approach provides insights into the conservation of residues and the functional importance of specific positions within the sequence.

To generate PSSMs, we employed Multiple Sequence Alignment (MSA) by running `PSI-BLAST` (Position-Specific Iterated BLAST), as described by Altschul et al. (1997). The specific parameters used for `PSI-BLAST` were `-num_iterations 3 -evalue 0.01`. The query sequence is searched through the `nrdb90` database (Holm and Sander, 1998) using the `PSI-BLAST` tool.

Given that the length of AMP sequences is typically less than 100, some peptides could not have their alignments generated using the nrdb90 database. For these peptides, PSSMs were produced by using PSI-BLAST to search the NR database (O'Leary et al., 2016) instead of the nrdb90 database.

The dimension of the PSSM feature is $L \times 20$-D, where L is the length of the peptide sequence. Each entry in the PSSM represents the log-likelihood of a specific amino acid occurring at a given position, based on the observed frequencies from the MSA.

The PSSM is mathematically defined as:

$$\text{PSSM}_{ij} = \log\left(\frac{f_{ij}}{b_j}\right)$$

where: - PSSM_{ij} is the score for amino acid j at position i, - f_{ij} is the observed frequency of amino acid j at position i in the MSA, - b_j is the background frequency of amino acid j in the database.

By incorporating PSSMs into our peptide analysis, we leverage the evolutionary conservation patterns, enhancing the predictive power of our models. PSSMs provide a robust representation that captures the functional and structural importance of each residue in the peptide sequence.

4. **Physicochemical Properties:** We used 10 physicochemical properties to represent the characteristics of each amino acid:

   - Hydrophobicity (KD, HW, EI, RO)
   - Charge
   - Polarity
   - Polarizability
   - Van der Waals Volume
   - Isoelectric Point
   - Molecular Weight

3.4.2 Edge Features

1. **Distances:** The Euclidean distance between Cα atoms of residues, used as an edge feature to represent spatial proximity.

2. **Angles:** Three angles capturing the geometric relationships between residues, used as additional edge features.

Chapter 4

Experiments and Results

4.1 Dataset

The dataset used in this study is derived from the dbAMP (Database of Antimicrobial Peptides) database, a well-regarded source for antimicrobial peptides (AMPs). The dataset is a collection of AMP sequences that have been preprocessed to ensure data quality. We have used this data from [1]. The dataset includes peptides of varying lengths, but we only used sequences with more than 10 amino acids to ensure realistic modeling and better generalization.

4.1.1 Dataset Characteristics:

- **Total Number of Peptides:** 6,845 AMPs
- **Data Split:** The dataset is divided into a training set and an independent testing set using stratified random sampling to maintain the proportionate representation of each functional activity. Specifically, the training set contains 6,160 sequences, while the testing set includes 685 sequences.
- **Functional Classes:** The dataset includes peptides with various functional activities categorized into several classes:

- Antibacterial
- Targeting Mammalian Cells
- Antifungal
- Antiviral
- Anticancer

Functional class	**Training set**	**Independent testing set**
Antibacterial	4685	531
Targeting Mammalian Cells	2514	276
Antifungal	2007	223
Antiviral	977	102
Anticancer	814	90

Table 4.1: Number of peptides in each functional class

The focus is primarily on the antibacterial category, as it is more general and has more related data. Therefore, we prioritized this class in our study. Figure 4.1 illustrates the amino acid composition of peptides for different classes.

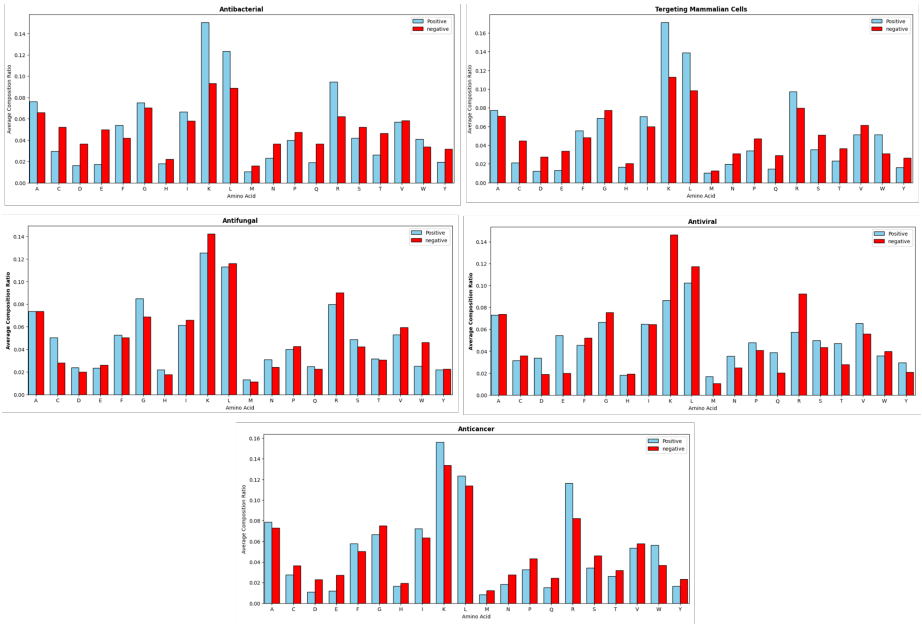


Figure 4.1: Average Composition Ratios for different classes

4.1.2 3D Structure Computation and Validation

As discussed in Section 3.3, we employed multiple computational tools to predict the 3D structures of peptides and validated these structures against experimentally determined models from the RCSB Protein Data Bank (PDB). The validation was conducted using both visual inspection and quantitative assessment via Root Mean Square Deviation (RMSD) calculations. Based on these evaluations, AlphaFold was selected as the primary tool due to its superior accuracy and consistency.

Figures 4.2 illustrates the visualization and validation process using a reference protein from RCSB PDB, whose experimentally determined 3D structure served as a ground truth model.

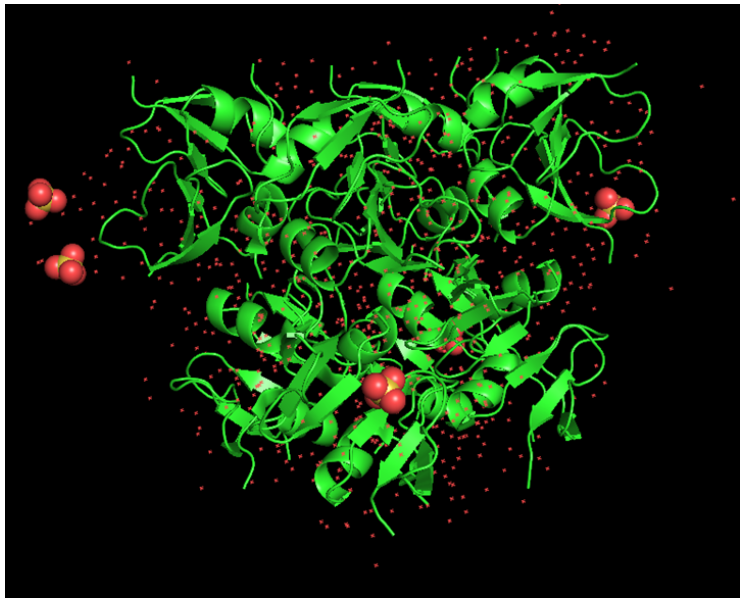
- *Top*: Displays the full 1F4D protein, including all its chains and molecular components.
- *Middle Left*: Shows the 3D geometry predicted by AlphaFold for the same protein.
- *Middle Right*: Depicts the alignment of our AlphaFold-generated structure with chain A of the experimental 1F4D protein, which corresponds to a peptide of interest.
- *Bottom*: Presents the alignment results, including the final RMSD value (0.437 Å), indicating a highly accurate structural match with minimal deviation.

As observed in the alignment visualization, the structures align with high precision, achieving an RMSD of less than 0.5 Å, which is considered excellent for structural similarity.

Given these results, AlphaFold was used to generate 3D structures for all training and test samples in our study. Since AlphaFold outputs the five best structural models for each sequence, we selected the top-ranked model for each peptide to construct the corresponding graph representation.

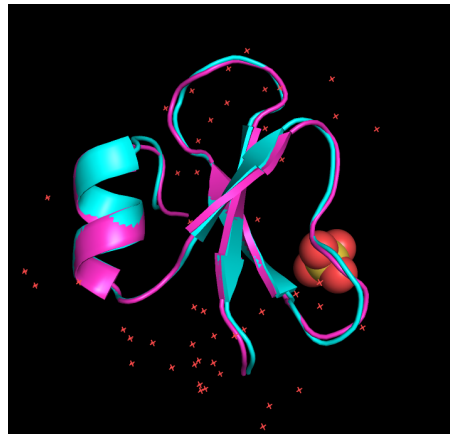
4.2 Models

To explore the capability of graph-based deep learning models in peptide functionality classification, we implemented and analyzed two primary Graph Neural



(a) Validated 1F4D protein structure

(b) AlphaFold predicted 3D structure



(c) Alignment visualization

```
MatchAlign: aligning residues (41 vs 41)...
MatchAlign: score 164.526
ExecutiveAlign: 298 atoms aligned.
ExecutiveRMS: 22 atoms rejected during cycle 1 (RMSD=1.07).
ExecutiveRMS: 15 atoms rejected during cycle 2 (RMSD=0.65).
ExecutiveRMS: 10 atoms rejected during cycle 3 (RMSD=0.51).
ExecutiveRMS: 5 atoms rejected during cycle 4 (RMSD=0.46).
ExecutiveRMS: 2 atoms rejected during cycle 5 (RMSD=0.44).
Executive: RMSD =    0.437 (244 to 244 atoms)
```

(d) Alignment result with RMSD score

Figure 4.2: Validation process using 1F4D protein: (a) Original structure, (b) AlphaFold prediction, (c) Structural alignment, (d) RMSD evaluation.

Network (GNN) architectures: the Graph Convolutional Network (GCN) and the Graph Attention Network (GAT). Our objective was to investigate their performance, particularly in classifying peptides into functional classes, with a strong emphasis on antimicrobial peptides (AMPs) due to their biomedical significance.

A key aspect of our study was to examine how GNNs perform using different feature representations, including basic sequence-based encodings and physicochemical properties, as well as to assess the impact of using geographical features and also incorporating edge attributes into the model. Since GCNs do not inherently support edge attributes while GATs do, this distinction enabled a comparative evaluation of how relational information within the peptide structure influences classification performance.

4.2.1 Computational Resources

All experiments were conducted on the DTU High-Performance Computing (HPC) Cluster, a powerful computational resource available to DTU researchers and students. This cluster provided access to multiple GPU nodes, enabling efficient model training and hyperparameter tuning.

4.2.2 Model Architectures

- **Graph Convolutional Network (GCN)**

  The architecture consisted of:

  - **Three graph convolutional layers**, each followed by layer normalization to stabilize training.
  - **ReLU activation functions**, applied after each layer to introduce non-linearity.
  - **Two fully connected (linear) layers**, applied after convolutional layers for final classification.
  - **Mean-pooling layer**, used to aggregate graph-level representations.

- **Graph Attention Network (GAT)**

  The architecture included:

  - **Three graph attention layers**, each using 8 attention heads to enhance learning efficiency.

- **Layer normalization**, applied to each layer to prevent gradient instability.
- **ReLU activation functions**, ensuring non-linearity in feature transformation.
- **Two fully connected (linear) layers**, serving as the final classification layers.
- **Mean-pooling layer**, aggregating node-level features into a graph-level representation.

4.3 Feature Representations and Attributes

Feature selection plays a critical role in determining the effectiveness of graph-based models. In this study, we explored two distinct feature sets:

- **One-Hot Encoding**

  One-hot encoding represents each amino acid in a binary vector format, preserving sequence identity while treating all amino acids as independent categories. This encoding is simple and ensures that peptide sequences are effectively transformed into a machine-readable format.

- **Comprehensive Feature Set**

  Beyond one-hot encoding, we experimented with a comprehensive 70-dimensional feature representation, which integrates a diverse range of biological and physicochemical properties. This feature set included:

  - **One-Hot Encoding**: Basic categorical representation of amino acids.
  - **Positional Encoding**: Captures sequential positioning of amino acids within peptides.
  - **Position-Specific Scoring Matrix (PSSM) Encoding**: Incorporates evolutionary information derived from sequence alignments.
  - **Hydrophobicity and Hydrophilicity Scores**: Quantifies the tendency of amino acids to interact with water or lipid environments.
  - **Charge and Isoelectric Point (pI)**: Determines the overall charge distribution of the peptide.
  - **Molecular Weight**: A key structural property influencing peptide interactions.
  - **Aliphatic Index**: Measures the stability of peptides in different conditions.

  - **Polarizability and Electrostatic Properties**: Defines interaction potential with biomolecules.
  - **Flexibility and Surface Accessibility**: Important for understanding binding efficiency.

This expanded feature set aimed to capture a broader spectrum of peptide characteristics.

4.4 Training and Evaluation

Each model was trained for 250 epochs, and model performance was assessed using the following metrics:

4.4.1 Models Performances

The performance of the GCN and GAT models was evaluated using the following metrics:

- **Accuracy**: The proportion of correctly predicted instances among the total instances.
- **Precision**: The ratio of true positive predictions to the total predicted positives.
- **F1 Score**: The harmonic mean of precision and recall.
- **Area Under the ROC Curve (AUC)**: A measure of the ability of the model to distinguish between classes.

4.4.2 Hyperparameter Tuning

The best hyperparameter combination was selected based on performance metrics, and the chosen configuration was used to train the models for 250 epochs.

Hyperparameter	GCN Model	GAT Model
Learning Rate (lr)	0.001	0.001
Batch Size	32	64
Hidden Channels	32	32
Weight Decay	0.0005	0.0005

Table 4.2: Optimal hyperparameters for GCN and GAT models.

4.5 Results

Table 4.3 provides key classification metrics: accuracy, precision, F1-score, and AUC (Area Under the Curve). These metrics help assess the effectiveness of each model in distinguishing peptide functionalities, particularly focusing on antimicrobial properties.

Table 4.3: Comparison of Metrics for Different Models

Model	Accuracy	Precision	F1	AUC
GAT_70_Attr	0.8642	0.8904	0.9172	0.8244
GAT_70_noAttr	0.8730	0.8966	0.9218	0.8252
GAT_hot_noAttr	0.8774	0.9089	0.9238	0.8298
GCN_hot_noAttr	**0.8847**	**0.8176**	**0.9269**	**0.8619**
GCN_70_noAttr	0.8628	0.9104	0.9130	0.8472

The GCN with one-hot encoding and no edge attributes (GCN_hot_noAttr) achieved the highest accuracy (0.8847) and F1-score (0.9269), indicating strong performance on this dataset. This model also attained the highest AUC (0.8619), confirming its superior ability to distinguish between positive and negative peptide samples across different classification thresholds.

The one-hot encoding-based models (GCN_hot_noAttr and GAT_hot_noAttr) consistently performed well across all metrics, with GAT_hot_noAttr achieving the highest AUC (0.8298) among GAT models. In contrast, models incorporating 70 additional features showed mixed performance, suggesting that the additional feature set may not provide significant benefits in this classification task.

Training results (Figure 4.3 and Figure 4.4) show simple models without using features have faster loss decrease and accuracy increase. GCN with only hot encoding features reaches best accuracy. Models with features used are slower and have less accuracy.

Test results are shown in figures 4.5 and 4.6. (Baseline is the situation if we la-

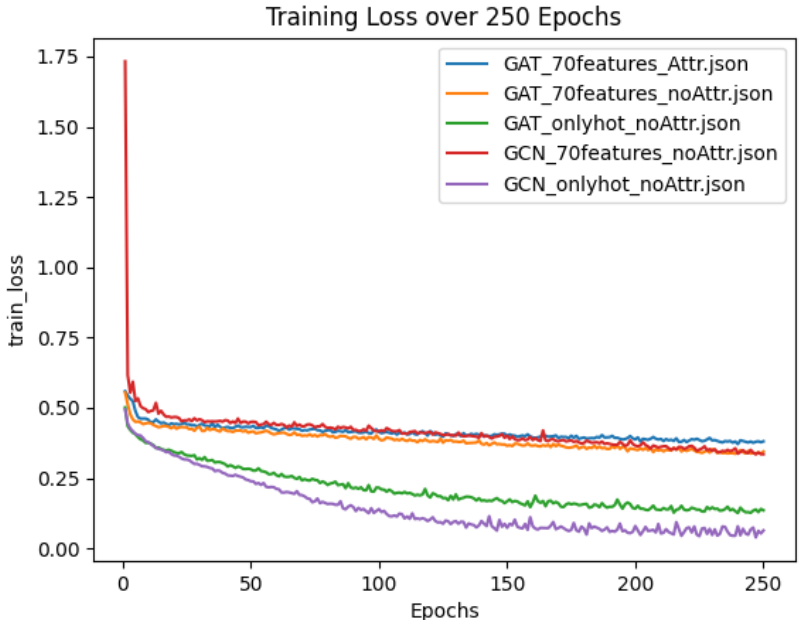


Figure 4.3: Comparison of Training Loss for Different Models.

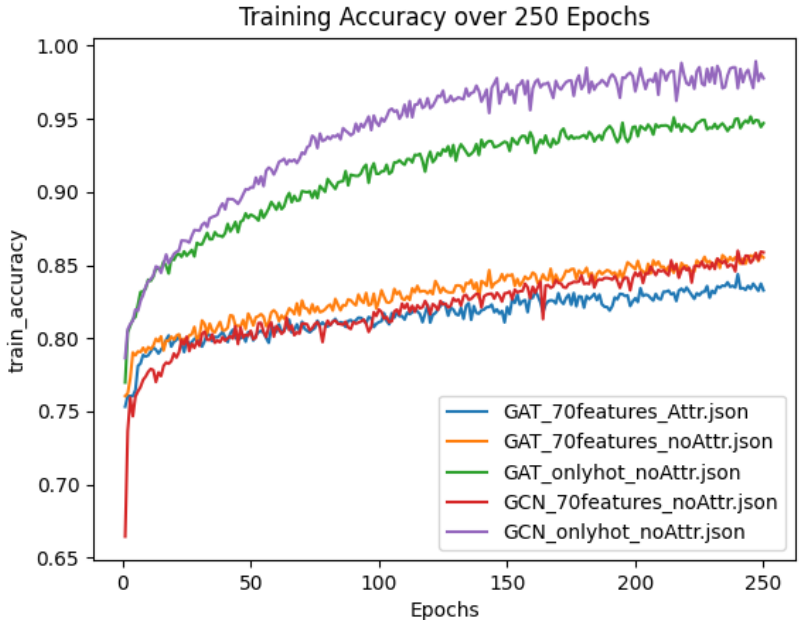


Figure 4.4: Comparison of Training Accuracy for Different Models.

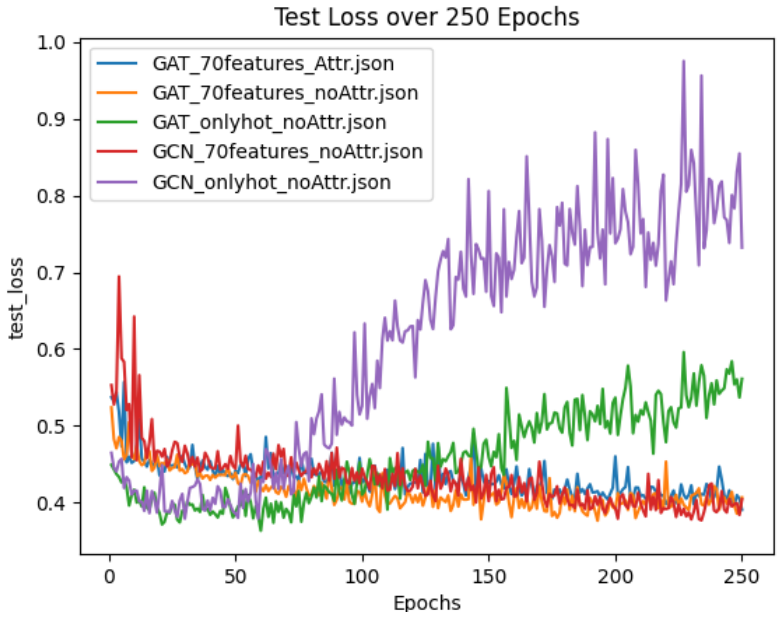


Figure 4.5: Comparison of Test Loss for Different Models.

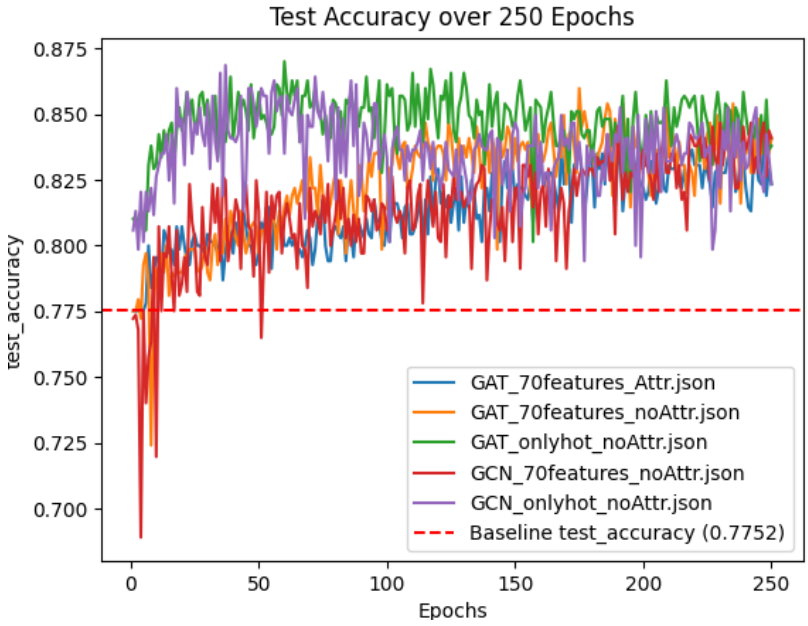


Figure 4.6: Comparison of Test Accuracy for Different Models.

bel all samples positive). The GCN model with only one-hot encoding (purple) shows significant instability in test loss and unstable accuracy despite high training accuracy, indicating overfitting. The loss increases as training progresses, suggesting the model is memorizing the training data but generalizing poorly. Also GAT model with only one-hot encoding (green) shows possible overfitting. It shows better stability than GCN but wothout more features it lacks generaliztion. The GCN model with 70 features (red) stabilizes at a lower test loss but exhibits fluctuations, suggesting some degree of overfitting but better generalization than the one-hot encoding version. The GAT models (blue and orange) show consistently lower test loss and more table accuracy, suggesting that the attention mechanism helps in learning more generalized representations and the feature-rich representations aid in generalization (particularly the version with additional features).

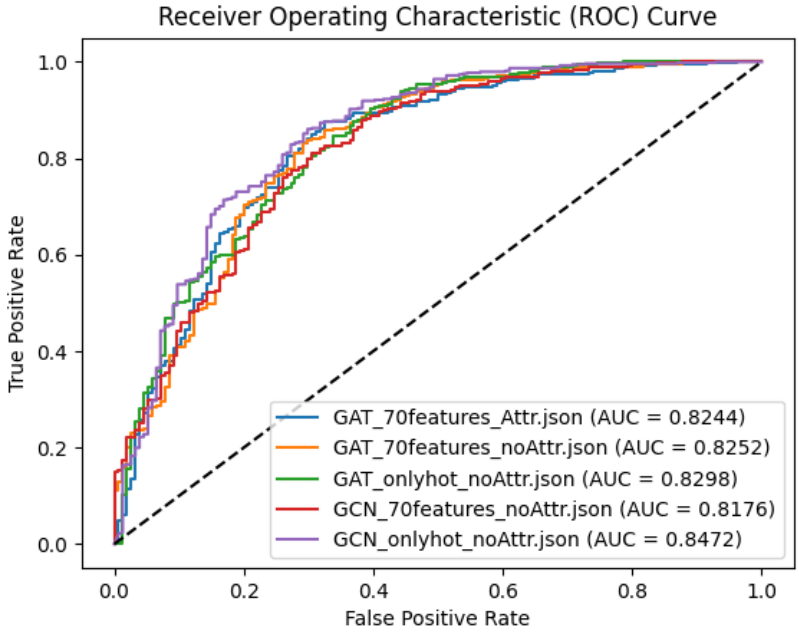


Figure 4.7: Receiver Operating Characteristic (ROC) Curves for Different Models.

ROC curve (Figure 4.7) show all models achieve AUC values above 0.81, demonstrating strong classification capability and overally, GAT and GCN models are competitive, with GCN showing slightly higher AUC but GAT maintaining more stable test accuracy and loss trends. Specifically, GCN with one-hot encoding

(purple) achieves the highest AUC (0.8472), suggesting that despite its overfitting issues in training, it still performs well in separating positive and negative classes. GAT models with additional features (blue and orange) perform slightly worse than the one-hot encoded GAT, suggesting that maybe adding more features does not necessarily improve AUC performance.

Figure 4.8 shows performance of the best model for classification of all classes of data, shows acceptable performance for antibacterial, antiviral and anticancer, but weaker on other two classes.

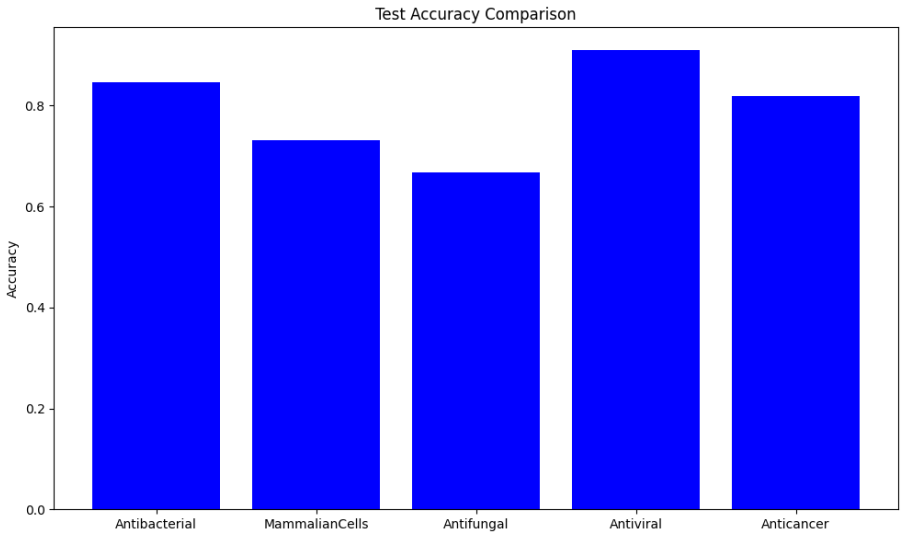


Figure 4.8: Accuracy of the GAT model on the different classes

4.5.1 Evaluation on the Arla Dataset

This project was conducted as a collaboration between DTU and Arla Food Ingredients, with a primary focus on identifying and classifying antibacterial peptides. After selecting the best-performing model based on its accuracy and generalization capability, we proceeded to evaluate its effectiveness on an independent dataset provided by Arla.

The Arla dataset consists exclusively of antibacterial peptides, which are of significant interest to the company. It contains a total of 186 peptide sequences of varying lengths, ranging from 3 to 344 amino acids. Since all sequences in this dataset belong to the antibacterial class (i.e., positive samples), we introduced

156 non-antibacterial peptides from our original test dataset to ensure a more balanced evaluation. This step was essential to prevent bias and enable a more realistic assessment of the model's discriminative power.

To maintain consistency with our previous experiments, we followed the same preprocessing pipeline. This evaluation aimed to validate the robustness of our model in real-world applications by assessing its ability to distinguish antibacterial peptides from non-antibacterial ones in a completely new dataset. The results of this evaluation, along with performance metrics, are discussed in the following sections.

Table 4.4: Results of testing on Arla dataset

Metric	Value	
Loss	0.8346	
Accuracy	**0.7529**	All Positive Result: 0.5470
Precision	0.7713	
Recall	0.7796	
F1 Score	0.7368	
AUC	0.7980	

Chapter 5

Discussion

The results of this study highlight both the strengths and limitations of graph-based deep learning models in peptide classification, particularly in distinguishing antimicrobial peptides. By employing Graph Convolutional Networks (GCN) and Graph Attention Networks (GAT), we aimed to understand how different graph neural architectures, feature representations, and structural attributes affect classification performance. While our models achieved reasonable classification accuracy, several challenges emerged, particularly regarding generalization, data limitations, and structural representation, all of which warrant further discussion.

5.1 Overfitting and Generalization Challenges

One of the most significant observations was the **trade-off between accuracy and generalization**. The GCN model with only one-hot encoding achieved the highest training accuracy, reaching nearly perfect classification on the training set. However, this model exhibited severe overfitting, as seen in its unstable test loss and fluctuating test accuracy. This suggests that while the model effectively learned patterns in the training data, it struggled to generalize to unseen sequences. Overfitting was particularly evident in the increasing test

loss during later epochs, indicating that the model memorized training data rather than learning meaningful, generalizable features.

The issue of overfitting was less pronounced in GAT models, which demonstrated more stable test performance. This could be attributed to their attention mechanisms, which enable the model to focus on more relevant node interactions rather than blindly learning correlations. Additionally, models trained with a more comprehensive feature set, while not significantly improving accuracy, exhibited **greater stability** in test performance, suggesting that incorporating physicochemical properties helped mitigate overfitting to some extent.

5.2 Limitations of the Dataset

While the dataset used in this study provided a valuable resource for antimicrobial peptide classification, it also had several limitations that likely affected model performance:

- **Insufficient dataset size**: Despite using a well-curated dataset, the total amount of data available may still not be sufficient for training highly complex models like GNNs. Deep learning models, especially those utilizing graph structures, often require large-scale data to fully capture meaningful patterns and avoid overfitting. A larger and more diverse dataset would likely improve model generalization and robustness.
- **Limited sequence diversity**: One possible reason for the model not achieving higher performance could be the presence of high sequence similarity within the dataset. Many peptide sequences may share common substructures, meaning that minor variations—such as adding or removing one or two amino acids—do not necessarily create functionally distinct sequences. For instance, a sequence like ABCDE may not be significantly different from ABCDEF, yet the model is expected to classify them distinctly. If the dataset contains a large number of such similar sequences, it may reduce the model's ability to learn truly discriminative features, making generalization more challenging.
- **Imbalance in class representation**: Although we applied stratified sampling, some classes (e.g., antibacterial peptides) were significantly more prevalent than others. This could have led to model bias toward the overrepresented classes, reducing its ability to generalize to less common functional groups. A more balanced dataset or data augmentation strategies could help mitigate this issue.

Despite these limitations, the dataset still provided valuable insights into the applicability of GNN-based classification models.

5.3 Impact of Feature Representation

The evaluation of different feature representations provided important insights:

- **Minimal gains from additional features**: Despite initial expectations that incorporating physicochemical properties, evolutionary scores, and structural attributes would enhance classification performance, the results suggested otherwise. The models trained solely on one-hot encoding performed comparably, if not better, than those using additional features. This suggests that:
  - The added features did not introduce significant new information beyond what the sequence alone provides.
  - The expanded feature space may have introduced additional complexity, making it harder for the model to learn efficiently.
  - While features like charge, hydrophobicity, and molecular weight are biologically relevant, they may not have been optimally utilized in the graph-based learning framework.
- **Reduction in overfitting**: One notable advantage of using additional features was a reduction in overfitting. While one-hot encoding models showed high variance between training and test results, those incorporating physicochemical features exhibited more stable learning curves, suggesting that even if the accuracy gains were minimal, the model's robustness and stability improved.

5.4 Challenges in 3D Structure Generation and Graph Construction

A major aspect of this study involved generating 3D structural representations of peptides using AlphaFold and constructing graphs based on these structures. While AlphaFold provides highly accurate structure predictions, several limitations emerged:

- **Limited structural validation**: Due to the scarcity of experimentally validated peptide structures, we were unable to directly validate the majority of our computationally generated 3D structures. Instead, we used a small set of validated peptides and proteins, such as 1F4D, to compare different structure prediction tools and determine the most reliable one. While AlphaFold was selected based on its strong alignment with these reference structures, its accuracy for other peptides remains uncertain, as no direct validation was possible for most sequences in our dataset.
- **Lack of experimentally validated structures**: Since experimentally determined peptide structures are scarce, computationally predicted structures had to be used for graph construction. While AlphaFold is highly accurate, it remains an approximation, and the absence of experimentally validated structures for most peptides prevented a comprehensive assessment of its reliability across all cases. We were only able to evaluate AlphaFold's predictions against a limited number of experimentally determined peptide structures.

5.5 Edge Attributes

Another key observation was the role of edge attributes in graph neural networks. Since GCN does not support edge attributes, its performance was driven primarily by node-level features and connectivity. On the other hand, GAT, which utilizes edge attributes, did not show substantial improvements in classification accuracy. This suggests that for this particular peptide classification task, the additional relational information provided by edges was not as crucial as initially expected. A possible explanation is that GNNs already use geometric features, so adding these attributes may not add more useful information.

However, future work could explore whether more sophisticated edge features—such as residue interaction energies or secondary structure constraints—could enhance performance.

5.6 Future Work

While this study demonstrates the potential of graph-based neural networks in peptide classification, several areas for improvement remain. Future research should focus on addressing the following key limitations to enhance model performance and generalization:

- **Expanding and improving the dataset:** One of the most significant limitations of this study is the relatively small dataset size. Increasing the number of peptide samples, particularly underrepresented functional classes, would help improve model robustness and generalization. Additionally, incorporating more diverse peptide sequences with better functional annotations can lead to more discriminative feature learning.

- **Enhancing 3D structural data:** The scarcity of experimentally validated peptide structures limits the reliability of computationally generated models. A larger collection of validated 3D structures would allow for better benchmarking and model evaluation. Furthermore, advances in computational structure prediction—such as the release of AlphaFold 3 (Which we could not use because of time and technical considerations) and other emerging techniques—may provide more accurate peptide geometries. Future studies should explore how improved structure prediction impacts graph-based learning approaches.

- **Hybrid modeling approaches:** While graph neural networks (GNNs) excel in capturing relational information, they may not fully exploit sequence-level features. Recent research suggests that combining GNNs with transformer-based models—which are particularly effective at modeling sequence dependencies—could lead to improved performance. Approaches such as Graph-Transformer hybrids or attention-based fusion of sequence encodings with structural graphs have shown promise in related fields and could be explored for peptide classification.

- **Alternative graph representations:** This study constructed peptide graphs based on C atom distances. Future work could investigate alternative graph representations that incorporate molecular interactions, hydrogen bonding networks, or secondary structure constraints to better capture functional properties. Additionally, learning-based graph construction methods, where edges and node connections are dynamically optimized, may further enhance classification accuracy.

By addressing these challenges, future studies can improve the reliability of graph-based learning in peptide classification and expand its applicability in computational biology and drug discovery.

Bibliography

[1] Y. Yan, H. Chen, and L. Xu, "Multi-label classification and features investigation of antimicrobial peptides with various functional classes,"

[2] Søren Drud-Heydary Nielsen, Ningjian Liang, Harith Rathish, Bum Jin Kim, Jiraporn Lueangsakulthai, Jeewon Koh, Yunyao Qu, Hans-Jörg Schulz & David C. Dallas, "Bioactive milk peptides: an updated comprehensive overview and database," *Critical Reviews in Food Science and Nutrition*, DOI: 10.1080/10408398.2023.2240, 2023.

[3] J. Xu, Z. Li, Y. Zhang, and J. Zhou, "Comprehensive assessment of machine learning-based methods for predicting antimicrobial peptides," *Briefings in Bioinformatics*, 2021. *Bioinformatics*, vol. 37, no. 5, pp. 640-648, 2021.

[4] C. D. Fjell, R. E. W. Hancock, and A. Cherkasov, "AMPer: a database and an automated discovery tool for antimicrobial peptides," *Bioinformatics*, vol. 23, no. 9, pp. 1148-1155, 2007.

[5] P. K. Meher, T. K. Sahu, V. Saini, et al., "Predicting antimicrobial peptides with improved accuracy by incorporating the compositional, physicochemical and structural features into Chou's general PseAAC," *Scientific Reports*, vol. 7, p. 42362, 2017.

[6] T. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," *International Conference on Learning Representations (ICLR)*, 2017.

[7] C. R. Chung, T. R. Kuo, L. C. Wu, et al., "Characterization and identification of antimicrobial peptides with different functional activities," *Briefings in Bioinformatics*, vol. 21, pp. 1098–1114, 2019.

[8] S. Thomas, S. Karnik, R. S. Barai, et al., "CAMP: a useful resource for research on antimicrobial peptides," *Nucleic Acids Research*, vol. 38, pp. D774–D780, 2010.

[9] S. Joseph, S. Karnik, P. Nilawe, et al., "ClassAMP: a prediction tool for classification of antimicrobial peptides," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 9, pp. 1535–1538, 2012.

[10] W. F. Porto, A. S. Pires, O. L. Franco, "CS-AMPPred: an updated SVM model for antimicrobial activity prediction in cysteine-stabilized peptides," *PLoS One*, vol. 7, p. e51444, 2012.

[11] H. Fu, Z. Cao, M. Li, et al., "Prediction of anuran antimicrobial peptides using AdaBoost and improved PSSM profiles," in *Proceedings of the Fourth International Conference on Biological Information and Biomedical Engineering*, Chengdu, China: Association for Computing Machinery, 2020, pp. 1–6.

[12] C. Li, D. Sutherland, S. A. Hammond, et al., "AMPlify: attentive deep learning model for discovery of novel antimicrobial peptides effective against WHO priority pathogens," *bioRxiv*, 2020. doi: 10.1101/2020.06.16.155705

[13] X. Su, J. Xu, Y. Yin, et al., "Antimicrobial peptide identification using multi-scale convolutional network," *BMC Bioinformatics*, vol. 20, p. 730, 2019.

[14] K. Yan, J. Chen, and Y. Zhang, "sAMPpred-GAT: prediction of antimicrobial peptide by graph attention network and predicted peptide structure," *Bioinformatics*, vol. 36, no. 8, pp. 2393-2400, 2020.

[15] J. Waghu and K. Barai, "CAMP: Collection of Anti-Microbial Peptides," *Nucleic Acids Research*, vol. 44, pp. D1094-D1097, 2016.

[16] S. Pirtskhalava, A. Gabrielian, and D. Cruz, "dbAMP: An Integrated Resource for Exploring Anti-Microbial Peptides," *Nucleic Acids Research*, vol. 44, pp. D1101-D1107, 2016.

[17] P. Meher, M. Sahu, and D. Saini, "iAMP-2L: A Two-Level Classifier for Identifying Antimicrobial Peptides and Their Functional Types," *Scientific Reports*, vol. 7, no. 1, p. 111, 2017.

[18] Y. Wang, T. Wang, S. Li, et al., "Enhancing geometric representations for molecules with equivariant vector-scalar interactive message passing," *Nature Communications*, vol. 15, p. 313, 2024.

[19] Z. Wang, Y. Wang, and X. Wu, "CS-AMPPred: A Method for Predicting Antimicrobial Peptides Based on Sequence Composition and Physicochemical Properties," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 17, no. 1, pp. 263-269, 2020.

[20] A. Thomas, B. Maiti, and J. Shankar, "Physicochemical Properties of Antimicrobial Peptides," *Journal of Theoretical Biology*, vol. 455, pp. 127-138, 2018.

[21] R. Veltri, J. Kamal, and S. Rosen, "Deep-AmPEP30: Deep Learning for Antimicrobial Peptide Prediction," *IEEE Access*, vol. 8, pp. 73045-73052, 2020.

[22] K. Fjell, J. Hancock, and R. Cherkasov, "AMP Scanner v.2: Predicting Antimicrobial Peptides Using Long Short-Term Memory Networks," *Bioinformatics*, vol. 32, no. 24, pp. 3815-3822, 2016.

[23] D. Lai, J. Chen, and J. Liu, "AMPlify: A Deep Learning Approach to Antimicrobial Peptide Prediction with Bi-LSTM and Attention Mechanisms," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 18, no. 1, pp. 293-300, 2021.