

DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

**PREDICTION OF PROTEIN-PROTEIN
INTERACTIONS THROUGH SEQUENCE BASED
CONTRASTIVE REPRESENTATION LEARNING
METHOD**

by
Duygu GEÇKİN

July, 2023

İZMİR

PREDICTION OF PROTEIN-PROTEIN INTERACTIONS THROUGH SEQUENCE BASED CONTRASTIVE REPRESENTATION LEARNING METHOD

**A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for Master of Science in
Department of Biomedical Technologies**

**by
Duygu GEÇKİN**

July, 2023

İZMİR

THESIS EXAMINATION RESULT FORM

We have read the thesis entitled “**PREDICTION OF PROTEIN-PROTEIN INTERACTIONS THROUGH SEQUENCE BASED CONTRASTIVE REPRESENTATION LEARNING METHOD**” completed by **DUYGU GEÇKİN** under supervision of **ASSOC.PROF.DR. GÜLESER KALAYCI DEMİR** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

.....
Assoc.Prof.Dr. Güleser KALAYCI DEMİR

Supervisor

.....
Assoc. Prof. Dr. Damla GÜRKAN KUNTALP

Jury Member

.....
Assist. Prof. Dr. Nalan ÖZKURT

Jury Member

Prof. Dr. Okan FISTIKOĞLU

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENT

First, I would like to express my deepest gratitude to my supervisor, Assoc. Prof. Dr. Güleser Kalaycı Demir for her continuous support, guidance, patience and kindness throughout my Master's education. Without her expertise, profound knowledge, the completion of this thesis would not have been possible.

I am also deeply grateful to Prof. Dr. Aydın Akan and Assoc. Prof. Dr. Osman Doluca for their guidance, and motivation. Their rich academic experience and valuable comments helped a lot in improving my research ability.

I also would like to thank to my colleagues RA Erdem Okur, RA Hande Aka Uymaz, RA Melek Büşra Temuçin, and RA Çınar Gedizlioğlu for their friendship and support.

I would also thank to Lecturer Oğuzhan Gökmen for being the most tolerant and understanding person throughout my most challenging moments and his precious help during my Master's education.

Last but certainly not least, I would like to thank my family for their endless love, encouragement, and support throughout my life.

Duygu GEÇKİN

PREDICTION OF PROTEIN-PROTEIN INTERACTIONS THROUGH SEQUENCE BASED CONTRASTIVE REPRESENTATION LEARNING METHOD

ABSTRACT

Proteins engage in protein-protein interactions (PPIs), forming physicochemical connections in order to carry out specialized functions such as transmission of signals, immune response, and metabolic cycles. These interactions hold great potential for applications in drug development, therapy design, and disease mechanism research. However, experimental methods for identifying PPIs are time-consuming, expensive, and limited in coverage. Therefore, computational methods are required to complement and enhance PPI identification. The presented study was aimed to construct PPI deep learning network utilizing only protein sequence information. Initially, protein sequences were encoded using three distinct methods: Binary Encoding, Autocovariance, and Position Specific Scoring Matrix (PSSM). Subsequently, a convolutional Siamese neural network was implemented to predict PPIs. The network architecture consisted of two duplicate subnetworks that used identical parameters to detect complicated patterns between pairs of protein sequences. The results demonstrated that the PSSM encoding method yielded the highest prediction accuracy, achieving 0.8407.

Since different encoding methodologies capture diverse features from the same protein sequence, an ensemble method was proposed to aggregate the outputs of these encoders, thereby improving prediction performance. As a result, the prediction accuracy increased to 0.8627. This highlights the efficacy of integrating multiple encoding strategies to harness their complementary strengths and enhance the accuracy of PPI prediction. The comprehensive experimental results prove that the proposed method is an effective, robust and reliable alternative approach for prediction of PPIs.

Keywords: Protein–protein interaction, Siamese neural network, Position Specific Scoring Matrices, Autocovariance, Binary representation

SEKANS TABANLI AYRIMSAL TEMSİL ÖĞRENMESİ YÖNTEMİ İLE PROTEİN-PROTEİN ETKİLEŞİMLERİNİN TAHMİNİ

ÖZ

Proteinler, sinyal iletimi yapmak, bağışıklık tepkisi oluşturmak ve metabolik döngüler yaratmak gibi spesifik işlevleri yerine getirmek için fiziko-kimyasal bağlantılar oluşturarak protein-protein etkileşimlerini meydana getirirler. Bu etkileşimler, hastalık mekanizmalarının incelenmesi, ilaç ve tedavi tasarımı gibi uygulamalar için büyük potansiyele sahiptir. Bununla birlikte, protein-protein etkileşimlerini belirlemeye yönelik deneysel yöntemler zaman alıcı, pahalı ve kapsamı sınırlıdır. Bu nedenle, protein-protein etkileşimlerinin tanımlaması ve buna yönelik çalışmaları geliştirmek için hesaplama yöntemlerine ihtiyaç duyulmaktadır. Sunulan çalışmanın amacı, proteinlerin sadece sekans bilgilerini kullanarak protein-protein etkileşimlerini derin öğrenme ağı oluşturarak kestirim yapmaktır. İlk olarak, protein sekansları üç farklı yöntem kullanılarak temsil edilmişlerdir: Binary kodlama, otokovaryans, ve Position Specific Scoring Matrix (PSSM). Daha sonra, protein-protein etkileşim kestirimi yapmak için evrişimli bir Siyam sinir ağı oluşturulmuştur. Ağ mimarisi, aynı parametreleri paylaşan iki özdeş alt ağdan oluşmaktadır ve protein dizisi çiftleri arasındaki karmaşık modellerin saptanmasını sağlamaktadır. Deneysel sonuçlar, PSSM kodlama yönteminin 0,8407 ile en yüksek tahmin doğruluğunu sağladığını göstermiştir. Farklı kodlama metodolojileri, aynı protein sekansında farklı özellikleri yakaladığından, her bir kodlayıcının Siyam sinir ağından çıkan tahmin değerleri toplanarak bir topluluk öğrenmesi yöntemi kullanılarak, tahmin performansı iyileştirilmiştir. Sonuç olarak, tahmin doğruluğu yüzde 0,8627'ye yükselmiştir.

Bu sonuçlara göre, protein-protein etkileşim tahmininin doğruluğunu artırmak için çoklu protein kodlama stratejilerini entegre etmenin etkinliğinin önemli olduğu görülmektedir. Kapsamlı deneysel sonuçlar, önerilen yöntemin protein-protein etkileşim tahmini için etkili, sağlam ve güvenilir bir alternatif yaklaşım olduğunu kanıtlamaktadır.

Anahtar kelimeler: Protein-protein etkileşimleri, Siyam sinir ağları, Position Specific Scoring Matrices, otokovaryans, Binary kodlama

CONTENTS

	Page
THESIS EXAMINATION RESULT FORM.....	ii
ACKNOWLEDGMENT	iii
ABSTRACT	iv
ÖZ.....	v
CONTENTS	vi
LIST OF FIGURES.....	viii
LIST OF TABLES	ix
ABBREVIATIONS.....	xi
 CHAPTER ONE INTRODUCTION.....	 1
1.1 Protein.....	1
1.1.1 Primary Structure.....	2
1.1.2 Secondary Structure.....	3
1.1.3 Tertiary Structure.....	3
1.1.4 Quaternary Structure.....	3
1.2 The Significance of Protein-Protein Interaction Detection.....	3
1.2.1 PPI Detection Methods	4
1.2.1.1 Experimental Methods for PPI Detection.....	4
1.2.1.2 Computational Methods for PPI Detection.....	5
1.3 Review on Prediction of PPI using only Sequence Information	7
1.4 Contributions of the Thesis.....	7
 CHAPTER TWO MATERIALS AND METHODS.....	 9
2.1 Dataset	9
2.2 Protein Sequence Encoding	10
2.2.1 Binary Representation.....	10
2.2.2 Autocovariance (AC).....	11
2.2.3 Position Specific Scoring Matrix (PSSM).....	13

2.3	Convolutional Neural Network (CNN).....	16
2.3.1	Learning Rate.....	16
2.3.2	Batch Size	17
2.3.3	Number of Filters.....	17
2.3.4	Kernel Size.....	17
2.3.5	Activation Function	17
2.3.6	Number of Layers	17
2.3.7	Padding	18
2.3.8	Stride Sizes	18
2.3.9	ReLU (Rectified Linear Unit) Layer	18
2.3.10	Pooling.....	18
2.4	Siamese Neural Network	18
2.5	Architecture of the Deep Learning Model	19
CHAPTER THREE EXPERIMENTS AND RESULTS.....		21
3.1	Evaluation Metrics	21
3.2	The Siamese CNN Method's Performance with Binary Encoding.....	22
3.3	The Siamese CNN Method's Performance with AC Encoding	26
3.4	The Siamese CNN Method's Performance with PSSM Encoding.....	29
3.5	Performance Comparison of Three Protein Encoding Method	33
3.6	Comparative Analysis of the Prediction Performances	35
3.7	Ensemble.....	39
CHAPTER FOUR CONCLUSION		44
REFERENCES		46

LIST OF FIGURES

	Page
Figure 1.1 Four levels of protein structure	2
Figure 2.1 Binary encoding of amino acids	11
Figure 2.2 Convolutional Siamese neural network framework.....	20
Figure 3.1 The effect of the neuron count in the FCL on the accuracy.....	27
Figure 3.2 Effect of the neuron count in the FCL	31
Figure 3.3 The framework for the proposed ensemble model	41
Figure 3.4 Ensemble network results with the different number of hidden neurons	43

LIST OF TABLES

	Page
Table 1.1 Amino acids & their abbreviations.....	1
Table 3.1 Number of protein couples in data sets	23
Table 3.2 Performance of the proposed network for different hyper-paramater sets of Binary Encoding on filtered dataset	25
Table 3.3 Hyper-parameters, activations, and output sizes for the Siamese CNN model for binary representation on a filtered dataset	25
Table 3.4 Number of protein couples in dataset for AC	26
Table 3.5 The effect of different layer configurations on the accuracy of AC method	28
Table 3.6 Performance of the proposed network for different hyper-paramater sets of AC on filtered dataset.....	32
Table 3.7 The effect of different configurations on the accuracy of PSSM.....	33
Table 3.8 Performance comparison of three encoding methods	35
Table 3.9 Performance comparison of different hyper-parameters on regular dataset using binary encoding	35
Table 3.10 Siamese CNN model layer configuration, hyper-parameters, and output sizes for binary encoding on regular dataset	36
Table 3.11 Comparison between our network and Richoux et al.'s fully connected model on regular dataset.....	37
Table 3.12 The number of interacted and non interacted protein pairs of strict test set.....	38
Table 3.13 Comparison between our network and Richoux et al.'s fully connected model on strict dataset	39

Table 3.14 Performance comparison of individual encoding methods using
proposed convolutional Siamese Network and ensemble method
combining Siamese and MLF-NNs on filtered dataset. 42



ABBREVIATIONS

PPI	: Protein-protein interaction
TAP	: Tandem affinity purification
AFM	: Atomic force microscopy
SVM	: Support vector machine
AC	: Autocovariance
kNN	: k-nearest neighbor
LD	: Local descriptor
RF	: Random forest
MCD	: Multi-scale continuous and discontinuous local descriptor
SAE	: Stacked autoencoder
CT	: Conjoint triad
DNN	: Deep neural network
APAAC	: Amphiphilic pseudo amino acid composition descriptor
CNN	: Convolutional Neural Network
MINT	: Molecular Interaction Database
BIND	: Biomolecular Interaction Network Database
DIP	: Database of Interacting Proteins
PSSM	: Position Specific Scoring Matrices
H ₁	: Hydrophobicity
H ₂	: Hydrophicity

V	: Volumes of side chains of amino acids
P ₁	: Polarity
P ₂	: Polarizability
SASA	: Solvent-accessible surface area
NCI	: Net charge index
SD	: Standard deviation
PSI-BLAST	: Position-specific iterated Basic Local Alignment Search Tool
FCL	: Fully-connected layer
ReLU	: Rectified Linear Unit
ACC	: Accuracy

CHAPTER ONE

INTRODUCTION

1.1 Protein

All living organisms are made up of living cells and share basic cellular mechanisms. These cell structures, whether belonging to bacteria or humans, contain the same building blocks of DNA, RNA and protein, although they differ in complexity. The sequences of the protein are synthesized by coding in DNA. The blueprint for each cell is encoded in the DNA sequence. The DNA is copied and part of it is transferred to mRNA, where it is converted into proteins (Ingber, 2000). Proteins are the most abundant organic molecules in the cell and make up most of the dry weight of the cell. Proteins are composed of long amino acid residues that form peptide bonds by binding to each other with covalent bonds. There are 20 types of amino acids in proteins, shown in Table 1.1 (Akram et al., 2011).

Table 1.1 Amino acids & their abbreviations

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

There are many different types of proteins in a cell. There are about five thousand types of proteins in the simplest bacterial cell, and each of these proteins has a different function (Thomas et al., 2003). Proteins play an essential roles in cellular processes,

such as the creation of signaling pathways, regulation of cellular processes, selective transport on the cell membrane, acceleration of chemical reactions. As a result, we can conclude that proteins are the fundamental building elements of cellular homeostasis. In many of these tasks, proteins work together by forming complexes of different sizes, replacing each other and transferring other proteins. They participate in the metabolism of the cell, signal transduction, DNA transcription, damage repair and many other mechanisms. Deciphering the exact functions of these protein complexes and determining the interactions between them is vital for understanding the functionality of a living cell (Smallridge, 2003).

The shapes of proteins are important for the realization of the function they will perform. How they transform into their unique final shapes is defined through four levels. These are primary, secondary, tertiary, and quaternary structures as seen in the Figure 1.1 (Hendy et al., 2015).

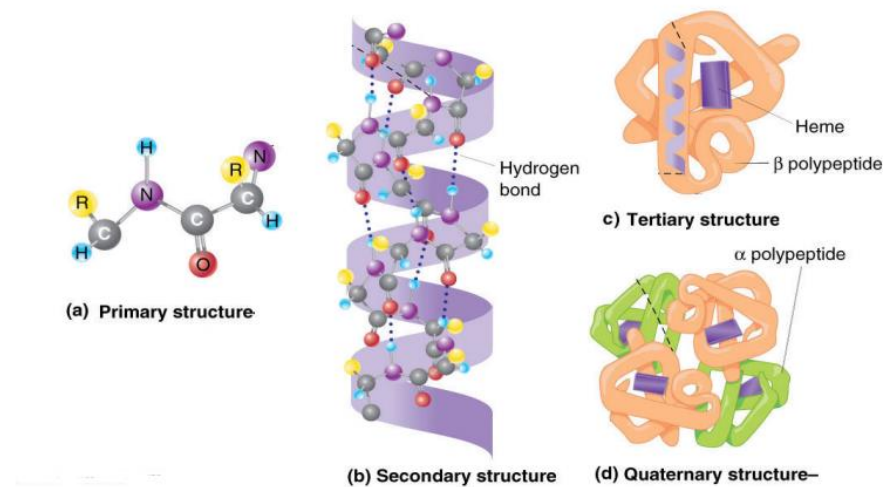


Figure 1.1 Four levels of protein structure (Peter J. Russell, 2009)

1.1.1 Primary Structure

The polypeptide chain, which is formed in the form of a sequence by binding amino acids to each other with peptide bonds, forms the primary structure of the protein. The representation of proteins in this way is called a protein sequence.

1.1.2 Secondary Structure

The secondary structure level pertains to the folding arrangement of the polypeptide backbone, which is stabilized by hydrogen bonds formed between the N-H and C=O groups. Numerous types of secondary structures have been identified, with the most prevalent ones being the α -helix and β -sheet. These structures exhibit a regular repeating pattern.

1.1.3 Tertiary Structure

The tertiary structure of a protein, after secondary structure formation of the polypeptide chain, further folds and twists in space with the sum of all the binding forces described previously, and the tertiary structure is formed.

1.1.4 Quaternary Structure

Proteins generally do not function as a single chain, although the tertiary structure shows the structural form for a single polypeptide chain. They are found in the form of a combination of polypeptides folded into more than one tertiary structure. The quaternary structure is formed when the polypeptides organize their own structures through their interactions with each other and come together (Mrozek et al., 2016).

1.2 The Significance of Protein-Protein Interaction Detection

When two or more proteins interact with one another, protein-protein interactions (PPIs) are formed. PPIs are key studies in biology as they are vital for numerous biological processes, including cell metabolism, signal pathways, DNA replication, and transcription (You et al., 2013). These interactions enable proteins to carry out their biological functions effectively. The prediction and detection of PPIs hold immense significance in the field of molecular biology and biomedical research. Understanding the interactions between proteins is crucial for deciphering the intricate mechanisms underlying cellular processes, signaling pathways, and disease development. It enables researchers to gain insights into protein networks, protein complexes, and the dynamics of cellular interactions. Understanding PPIs not only enhances our knowledge of biological processes but also plays a crucial role in identifying diseases and developing targeted treatments and therapies. PPI disruption

frequently results in diseases and has an important effect on the beginning or development of pathology (Kuzmanov & Emili, 2013).

Accurate prediction of protein-protein interactions aids in unraveling protein functions, identifying potential drug targets, and designing therapeutic interventions. Furthermore, the discovery of PPIs provides valuable insights into disease mechanisms and facilitates the detection of drug targets (Zahiri et al., 2013). By studying PPIs, we gain a deeper understanding of the intricate molecular networks underlying cellular functions and can develop innovative strategies to combat diseases (Jha et al., 2022).

Moreover, accurate predictions contribute to reducing experimental costs and time by prioritizing potential interactions for further experimental validation. By focusing on the prediction and detection of PPIs, scientists can advance our understanding of cellular processes and pave the way for the development of novel treatments and interventions for various diseases (Pan et al., 2016).

1.2.1 PPI Detection Methods

The detection of PPIs is essential for understanding the complex biological processes that occur within organisms. PPI research helps researchers understand how proteins interact to carry out particular activities and how these interactions might be altered or disturbed to cure diseases. Although the amount of data obtained by estimating PPIs using high-volume experiments is increasing day by day, there are significant difficulties in verifying these data using laboratory techniques.

Computational analysis of PPIs data is used extensively because it does not have difficulties such as experimental methods and provides information about the functions of proteins (Mahapatra & Sahu, 2021).

1.2.1.1 Experimental Methods for PPI Detection

PPI detection methods are categorically classified as experimental and computational methods (Chang et al., 2016). The experimental method can be divided into 2 types in vivo and in vitro (Rao et al., 2014). Studies using microorganisms, cells, or biological molecules in a laboratory setting are known as in vitro experiments. These studies, called test tube experiments, are carried out in a laboratory environment

with equipment such as test tubes, bottles, petri dishes and microtiterplates (Puig et al., 2001). In contrast to research that use tissue extracts or dead animals, in vivo investigations evaluate the impact of various biological agents on living creatures or cells. Animal tests and clinical studies are the main elements of in vivo research. In vivo tests are usually performed after in vitro tests because they are more suitable for observing the overall effects of an experiment on a living organism.

Various experimental techniques, both in vivo and in vitro, have been instrumental in investigating the conserved nature of protein interaction sites and screening numerous protein interaction partners. These techniques include Tandem affinity purification (TAP), nuclear magnetic resonance (NMR), atomic force microscopy (AFM), X-ray crystallography, chemical crosslinking, and so on (Zhu et al., 2019).

The identification of PPIs is only partially covered by biological experimental procedures because they are laborious and expensive. Consequently, there is a much greater need for development of computational methods (Yang et al., 2021).

1.2.1.2 Computational Methods for PPI Detection

Computational techniques usually based on different data types such as, the genomic information, protein structure and sequence information, co-evolutionary approaches, protein domains so on. Except for the sequence-based approach, these techniques can only be applied in limited circumstances as they rely on pre-knowledge about the protein. However, all proteins have a fundamental amino acid sequence, which is the most basic form of data that can be obtained from direct sequencing or translations of DNA sequences and contains a wealth of information (M. Chen et al., 2019).

With advancements in sequencing technology, there has been a significant accumulation of protein sequence data. Several studies have demonstrated that knowledge of amino acid sequences alone can be sufficient for the discovery of novel PPIs. Therefore, it is crucial to develop computational approaches that utilize protein amino acid sequence information for PPI prediction (Y. Li et al., 2022).

Several computational methods have been proposed for PPI prediction based on amino acid sequences, such as support vector machine (SVM) with autocovariance (AC) (Guo et al., 2008), k-nearest neighbor (kNN) with local descriptor (LD) (Zhou et al., 2011), SVM with conjoint triad method (Pnas, 2007), random forest (RF) with multi-scale continuous and discontinuous local descriptor (MCD) (You et al., 2014), Stacked Autoencoder (SAE) with AC and the conjoint triad (CT) (Sun et al., 2017), deep neural networks (DNNs) with amphiphilic pseudo amino acid composition descriptor (APAAC) (Chou, 2005).

Although these machine learning algorithms have shown success about PPI prediction using amino acid sequence information, they often encounter limitations in terms of memory and runtime. However, deep learning methods have gained significant attention due to their ability to address these challenges by drawing inspiration from the connections between neurons and learning processes observed in the human brain in various domains such as speech and image recognition (Krizhevsky et al., 2012), computer vision (Voulodimos et al., 2018) and natural language understanding (Lipton et al., 2015). They are capable of handling enormous quantities of data and automatically identifying useful aspects for objective tasks.

Deep learning architectures are now offering effective solutions for solving many bioinformatics issues including DNNs for predicting protein structures, Convolutional Neural Networks (CNNs) for analyzing DNA sequences, and hybrid convolutional and recurrent deep neural network for DNA function classification (Jumper et al., 2021).

Recently, two deep learning algorithms, ResNet50 and stacking autoencoder described by Jha and Saha (Jha & Saha, 2020) to extract features from protein sequence representations based on AC and conjoint triads. An LSTM-based classifier model was created for each feature encoding approach. The results of the experiment indicate that the deep learning algorithm was able to extract important features from the protein's multimodal data. The accuracy and effectiveness of PPIs prediction still need to be improved, even though computational-based approaches have made significant strides and show promising application potential, in order to serve as an additional tool for proteomics research and other bioinformatics requirements. (Y. Li et al., 2022).

1.3 Review on Prediction of PPI using only Sequence Information

Deep learning algorithms have recently been widely employed by academics to predict interaction labels based only on protein sequence data. Sun et al. performed the same task using a stacked autoencoder classifier. They obtain input representations for the classifier using the conjoint triad and the AC techniques for protein sequence coding. (Sun et al., 2017).

Typically, autoencoders convert input data to a compressed form in order to learn to reconstruct the input data. According to this theory, autoencoders discover nonlinear embeddings for the inputs that can accurately reflect the fundamental properties of the input data (Alain et al., 2014).

Siamese architectures, which are networks created to learn shared weights between the two examples in a pair, are typically used to represent paired data instances. A novel deep learning framework called DPPI was introduced by Hasfeminar et al. It accepts a pair of protein sequences and an empirically determined binding score as input. DPPI utilize a Siamese-like CNN architecture (Hashemifar et al., 2018).

Recently, a novel end-to-end framework called PIPR (Protein-Protein Interaction Prediction Based on Siamese Residual RCNN) has been introduced for PPI prediction solely based on protein sequences (M. Chen et al., 2019). In this framework, protein sequences are encoded using one-hot encodings of 15 hydrophobicity and electrostaticity classes. Additionally, pre-trained Skip-Gram embeddings are utilized to capture similarities in amino acid co-occurrence patterns.

Another important piece of work in this manner is DNN-PPI which uses two distinct CNN encoders and a different learning structure (H. Li et al., 2018). However, DNN-PPI does not use a Siamese learning architecture to completely characterize pairwise interactions of sequences, nor does it add physicochemical features into representations of amino acids (M. Chen et al., 2019).

1.4 Contributions of the Thesis

The objective of this study is to use a convolutional Siamese neural network to predict PPIs and to investigate the impact of various protein encoding techniques on

the accuracy of the predictions. Three different protein sequence encoding techniques presented: Binary Encoding, Autocovariance, and Position Specific Scoring Matrix.

The primary contributions of this thesis can be summarized in three key aspects.

The first contribution of this thesis involves the introduction of a convolutional Siamese neural network for predicting protein-protein interactions. This is achieved by employing three distinct protein sequence encoding methods: Binary Encoding, Autocovariance, and Position Specific Scoring Matrix. Notably, our approach differs from existing models by combining these encoding methods with Siamese-like architectures, which, to the best of our knowledge, has not been explored previously.

The second contribution of this thesis involves the development of an ensemble strategy aimed at enhancing the performance of a single predictor. This is achieved by aggregating the outputs of multiple protein encoders. By leveraging the collective insights from these diverse encoders, we were able to improve the predictive capabilities of the model, resulting in enhanced performance compared to using a single predictor alone.

The third contribution of this thesis involves the training of our network model using Richoux et al.'s regular set of data. However, to ensure a robust evaluation, as part of our methodology, we implemented a filtering mechanism for the test set, specifically designed to identify and remove mirror copies of proteins that were already present in the training set. This approach ensured that the test set remained independent and free from any potential bias or duplication caused by proteins appearing in both training and test sets. By removing these mirror copies, we enhance the robustness and validity of our evaluation process.

CHAPTER TWO

MATERIALS AND METHODS

2.1 Dataset

Several databases of PPIs have been established to date, including HAPPI database (J. Y. Chen et al., 2009), Molecular Interaction Database (MINT) (Licata et al., 2012), APID database (Alonso-López et al., 2019), Biomolecular Interaction Network Database (BIND) (Bader et al., 2001), and Database of Interacting Proteins (DIP) (Xenarios et al., 2002), as well as publicly accessible databases such as TAIR (Lamesch et al., 2012), BioGRID (Oughtred et al., 2019), and IntAct (Kerrien et al., 2012).

We used a dataset collected by the Richoux et al. (Richoux et al., 2019). A list of interacted protein pairs was extracted from the UniProt website, which was queried on June 18th, 2018 to retrieve all human protein sequences that showed evidence of interacting with other proteins. Using biopython, they created an internal python script to generate a negative dataset from a selection of randomly chosen proteins after first ensuring that no interaction existed. From this set, they removed proteins made up of no more than 1166 amino acids. They then randomly split this set into three separate groups, each of which had exactly 50% positive and negative samples. These three sets are the hold-out test set, the hold-out validation set, and the training set. We trained our network using Richoux’s medium train dataset that contains 26,303 interacted protein pairs and 26,303 non-interacted protein pairs.

Richoux’s test protein pairs that never appear into the training set however, their mirror couple were present. We removed the protein pairs with mirror couple from the test dataset in order to prevent overfitting of our convolutional Siamese network. This study utilized a total of 56,674 human protein interactions, with 52,606 interactions being used for training our network model and 797 interacted, 3271 non-interacted in total 4068 interactions used for testing.

2.2 Protein Sequence Encoding

Sequence-based protein encoding involves representing proteins based on the information derived from their amino acid sequences. Several encoding techniques have been developed to capture various aspects of protein sequences, enabling the prediction of PPIs. In many sequence-based prediction problems, feature encoding becomes a crucial and labor-intensive task to show the powerful capabilities of feature learning within deep learning models (H. Li et al., 2018).

In order to process amino acid sequences in a computational model, a vital step is to assign numerical representations to each amino acid. This process, known as numerical encoding, is commonly employed in various encoding approaches. Notably, even though they give a numerical representation of amino acid sequences, these encoding schemes usually try to capture prior domain information about amino acids as shared characteristics between vectors (N. Lin et al., 2004).

The majority of recently created models continue to employ encoding techniques that were created during the deep learning time frame. Examples include substitution matrices like the BLOSUM (Hess et al., 2016), orthogonal encoding, also known as “one-hot encoding” (K. Lin et al., 2002), and physicochemical character-based schemes such as AC (Guo et al., 2008).

The following approaches are used in this work to encode amino acid residues: Binary Representation (Richoux et al., 2019), AC (Guo et al., 2008), and Position Specific Scoring Matrices (PSSM) (An et al., 2019).

2.2.1 Binary Representation

Binary Encoding is a popular method for representing protein sequences, where each amino acid is encoded as a binary vector. The one-hot encoding, often known as orthogonal encoding, is the most widely used Binary Encoding. In this encoding scheme, each position in the vector corresponds to a specific amino acid, and a value of 1 is assigned to the position representing the amino acid present in the sequence, while all other positions are set to 0. For instance, the twenty standard amino acids are arranged as follows: [A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y], the

one-hot codes for A and C are respectively [10000000000000000000] and [01000000000000000000] (Jing et al., 2020). Figure 2.1 demonstrates how the one-hot encoding technique is utilized to represent amino acids within a protein sequence.

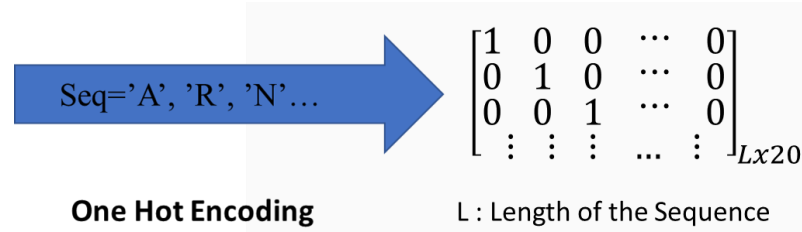


Figure 2.1 Binary encoding of amino acids

Our datasets also include selenocysteine (U), a placeholder for either asparagine or aspartic acid (B), another one for either glutamic acid or glutamine (Z), and a placeholder for an unidentified acid (X), in addition to the 20 standard proteinogenic amino acids. In a protein sequence, each amino acid is represented as a 24-long binary vector. To ensure that all encoded sequence matrices have the exact length required, zeros are added to the one-hot encoded information. This is necessary because the maximum supported sequence length is 1166 (Richoux et al., 2019).

This Binary representation provides a concise and efficient way to capture the presence or absence of specific amino acids in protein sequences. Binary Encoding offers a flexible and computationally efficient approach for representing protein sequences, enabling the application of multiple machine learning methods for PPI prediction.

2.2.2 Autocovariance (AC)

Statistics uses the term AC to describe the covariance of a certain time series or continuous signal X_t . Specifically, the linear statistical dependencies between a sequence's sample are expressed by the autocorrelation (or autocovariance) (J. Li & Chen, 2013).

In this study, AC used to encode the protein sequences. An examination of the correlation and interaction of the variables at different positions that will result in efficient method to predict PPIs (Sun et al., 2017).

AC dataset was created by removing pairs of protein with atypical amino acids, Z, U, B and X from the main dataset (Richoux et al., 2019), yielding 26,289 interacted and 26,206 non-interacted protein pairs.

There are three main steps in AC method to encode the protein sequences.

First, seven physicochemical properties of amino acids chosen to represent each amino acids in sequences. The original values of seven physicochemical characteristics are hydrophobicity (H_1) (TANFORD Vol & Charles Tanford, 1961), hydrophilicity (H_2) (Hopp & Woods, 1981), volumes of side chains of amino acids (V) (Imai et al., 2000) , polarity (P_1) & polarizability (P_2) (Chung et al., 2020), solvent-accessible surface area (SASA) (Durham et al., 2009) and net charge index (NCI) of side chains of amino acids (Fauchere' et al., 1988), respectively.

Second, Equation 2.1 was utilized to standardize these seven physicochemical data to zero mean and unit standard deviation (SD). P_{ij} is the j -th descriptor value for i th amino acid, P_j is the mean of the j th descriptor over the 20 amino acids and S_j is the corresponding standard deviation. To account for each protein sequence, the process involved the creation of seven distinct vectors. In these vectors, each amino acid was represented by a set of normalized values derived from seven descriptors. This comprehensive approach ensured a comprehensive representation of the amino acids within the protein sequence (Guo et al., 2008).

$$P'_{ij} = \frac{P_{ij} - P_j}{S_j} \quad (2.1)$$

$$(i = 1, 2, \dots, 20; j = 1, 2, \dots, 7)$$

Finally, the AC variables are calculated by using the Equation 2.2.

$$AC_{lag,j} = \frac{1}{n-lag} \sum_{i=1}^{n-lag} (X_{i,j} - \frac{1}{n} \sum_{i=1}^n X_{i,j}) (X_{(i+lag),j} - \frac{1}{n} \sum_{i=1}^n X_{i,j}) \quad (2.2)$$

The lengths of protein sequences differ from one another, which leads to unequal-length vectors. In order to use artificial intelligence methods like neural networks for training, a fixed number of inputs are necessary. AC, which is a statistical method was used to overcome the problem of unequal-length inputs.

Numerical sequences were converted into uniform matrices using AC (You et al., 2013). The primary focus of AC lies in assessing the neighboring effect, which pertains to the interactions among individual amino acids within a sequence, thereby enabling the identification of patterns that operate across the entirety of the sequences.

The AC variables refer to the average interactions between the amino acid units in a protein sequence that are separated by a particular distance, or “lag” throughout the entire sequence.

The lag value denotes to the distance from one amino acid to its neighbor residue. The descriptor variable j provides specific characteristics or features, while i corresponds to the position of the amino acid within the sequence denoted as 'X'. The variable 'n' represents the length of the sequence 'X', providing information about the total number of amino acids present in the sequence. By utilizing this representation, each protein sequence can be effectively captured and analyzed using the matrix representation (Guo et al., 2008).

2.2.3 Position Specific Scoring Matrix (PSSM)

As a result of ongoing biological evolution, one or more amino acid residues may be displaced, inserted, or deleted from protein sequences. As a result, proteins become less comparable with time. Homologous proteins may still share common structures, despite these modifications. PSSM was introduced to show how proteins exhibit this property (Y. Li et al., 2019).

PSSM is an evolutionary strategy built on feature extraction techniques that have proved effective in a number of bioinformatics disciplines, including the prediction of protein secondary structure, the identification of DNA-binding proteins and prediction of PPIs (Zahiri et al., 2013). Gribskov and colleagues introduced PSSM to find proteins having distant relationships for the first time in 1987 (Gribskov et al., 1987). Today, PSSM is widely utilized to identify protein folding patterns and proteins with distant relatives. In the field of proteomics, PSSM functions as a valuable scoring matrix that incorporates evolutionary information from protein sequences. Its widespread usage highlights its significance in protein analysis and characterization.

In this study, PSI-BLAST (position-specific iterated BLAST (Basic Local Alignment Search Tool)) (Altschul et al., 1997) application tool was employed in conjunction with the SwissProt database. This specific combination was employed to convert each protein sequence into a PSSM, which is a widely accepted numerical representation of protein sequences. This conversion was performed with the aim of facilitating subsequent tasks related to the detection of PPIs. The SwissProt database was chosen as the comparison database due to its meticulous verification process, which involves computer tools and access to pertinent literature. PSI-BLAST requires high sensitivity in sequence alignment between the input proteins and the proteins in the database to generate accurate PSSMs (Gao et al., 2016).

PSSM is a two-dimensional matrix that composed of L rows and 20 columns, where L represents the length of the protein sequence and 20 columns refers to the 20 naive amino acids (Y. Li et al., 2022). The components of $PSSM(i, j)$ are determined as shown in Equation 2.3.

$$PSSM(i, j) = \sum_{k=1}^{20} w(i, k) \times Y(j, k) \quad (2.3)$$

$$(i = 1, \dots, N ; j = 1, 2, \dots, 20)$$

The expression $w(i, k)$ denotes the rate of recurrence of the k -th amino acid, out of the 20 possible amino acids, at position i within functionally relevant protein sequences that have been aligned. This rate is calculated by comparing the number of occurrences of the k -th amino acid at position i with the total number of aligned protein sequences.

The Dayhoff's mutation matrix captures the relationship between different amino acids. Specifically, the differences the j -th and k -th amino acids in this matrix corresponds to the substitution matrix denoted as $Y(j, k)$. This substitution matrix, $Y(j, k)$, reflects the probability or rate at which one character (amino acid) in a protein sequence changes or is substituted by another character (amino acid) over a given period of time. It provides insights into the evolutionary dynamics and relationships between amino acids in protein sequences (Nanni et al., 2014).

Each matrix can be formulated as:

$$\text{PSSM} = \begin{bmatrix} P_{1,1} & P_{1,2} & P_{1,3} & \dots & P_{1,20} \\ P_{2,1} & P_{2,2} & P_{2,3} & \dots & P_{2,20} \\ \vdots & P_{i,j} & \vdots & \dots & \vdots \\ P_{L,1} & P_{L,2} & P_{L,3} & \dots & P_{L,20} \end{bmatrix} \quad (2.4)$$

The length of encoded sequences remains constant (1166). As a result, each protein sequence converted into 1166×20 matrices.

In this work, we execute PSI-BLAST using the MATLAB command line system, with the e-value set to 0.001 and the number of iterations set to 3, respectively.

The elements of the PSSM matrix typically include either positive or negative numbers. Specifically, the value of element $P_{i,j}$ in the i -th row of PSSM represents the probability of the i -th residue undergoing a mutation and being replaced by the j -th type among the 20 native amino acids. This probability is derived from the evolutionary information contained in multiple sequence alignments of the protein.

Positions exhibiting significant conservation receive elevated ratings, whereas positions demonstrating minimal conservation are assigned a score approaching zero (Gao et al., 2016).

2.3 Convolutional Neural Network (CNN)

CNNs, which are machine learning models introduced in the 1990s, are based on supervised learning. Similar to Artificial Neural Networks (ANNs), CNNs consist of an input layer, a hidden layer, and an output layer. The hidden layer in a CNN is responsible for extracting and categorizing features from the input data. This is achieved by applying a convolution operation with a weighted convolution filter to extract the features.

The pooling layer, which abstracts the input data, plays a crucial role in the CNN. There are two types of pooling: max pooling and average pooling. Unlike other layers, the pooling layer does not have trainable parameters and the number of channels remains constant. The fully-connected layer (FCL) utilizes a standard multilayer neural network design to classify the extracted features obtained in the previous feature extraction step.

One advantage of CNNs is their ability to directly process raw data without the need to pre-process and remove inherent properties of the data. This is achieved through the feature extraction stage of the CNN. The performance of a CNN is influenced by hyperparameters such as the size of the convolution kernel, number of channels, padding, stride, and the overall layer structure, including the size of the feature map. A suitable hyper-parameter should be employed in CNN to get the desired outcome (Lee et al., 2018).

Certainly, hyper-parameter tuning is a crucial step in optimizing the performance of CNNs. Here are some examples of hyper-parameters in CNNs and their tuning methods:

2.3.1 Learning Rate

This hyper-parameter controls the step size taken during the gradient descent optimization process. Step size refers to the model adaptation time and controlled by

the learning rate. A too high learning rate can cause the algorithm to diverge, while a too low learning rate can cause slow convergence. One common method for tuning the learning rate is to perform a grid search over a range of values.

2.3.2 Batch Size

This hyper-parameter controls the number of samples processed in each iteration during training. A too small batch size can cause unstable gradients, while a too large batch size can cause memory issues.

2.3.3 Number of Filters

This hyper-parameter controls the number of filters applied in each convolutional layer. A too low number of filters can cause underfitting, while a too high number of filters can cause overfitting. A common method for tuning the number of filters is to perform a grid search over a range of values.

2.3.4 Kernel Size

This hyper-parameter controls the size of the filter applied in each convolutional layer. A too small kernel size can cause loss of important features, while a too large kernel size can cause overfitting.

2.3.5 Activation Function

This hyper-parameter controls the nonlinearity applied to the output of each convolutional layer. A different activation function can result in different learning dynamics and performance. Common activation functions include ReLU (Rectified Linear Unit) Layer, sigmoid, and tanh.

2.3.6 Number of Layers

This hyper-parameter controls the depth of the neural network. A too shallow network can cause underfitting, while a too deep network can cause overfitting.

A hyper-parameter known as receptive field will be computed as a result of the convolution operation on the neurons associated to local regions in the input. The vectors that we acquire after convolution from each filter are feature vectors. The

following are the additional hyper-parameters that affect the convolutional layer's output size:

2.3.7 *Padding*

Padding is the process of extending the input's boundary with zeros to maintain the input's spatial size.

2.3.8 *Stride Sizes*

The arrangement of width and height dimensions is controlled by stride. Small stride sizes result in huge output. The receptive fields will overlap less with larger strides, resulting in output with smaller dimensions.

2.3.9 *ReLU (Rectified Linear Unit) Layer*

In order to apply non-linearity to the entire network without affecting the receptive fields of the convolutional layer, it provides efficient and simpler calculation and is applied element-wise.

2.3.10 *Pooling*

The pooling procedure reduces the representation's spatial size, which helps to reduce computations and parameters and thereby controls over-fitting. Max-pooling, which is effectively a non-linear down-sampling process along the spatial dimensions, is applied over the output.

2.4 Siamese Neural Network

The Siamese neural network, also known as the twin neural network, is a type of neural network architecture that is frequently used for tasks that involve determining how similar or dissimilar two inputs are, such as: recognizing similar handwritten character images, face recognition, and automatic identification of objects. Siamese neural network suggested as a solution to the image matching problem of signature verification in the 1990s (BROMLEY et al., 1993).

Each input in a standard Siamese neural network is routed through its own subnetwork, which is made up of one or more layers of neurons. Unlike earlier

architectures, the weights of these subnetworks are shared, allowing simultaneous updating of the parameters of both networks (W. Chen et al., 2022). To assess the similarity between the two inputs, the outputs of the respective subnetworks are compared using a distance metric, such as Euclidean distance or cosine similarity.

One common application of Siamese neural networks is in face recognition. In this scenario, each input corresponds to a different image of a face, and the network is trained to determine whether the two images depict the same person or different people. The network is trained using pairs of images, with each pair labeled as either "same" or "different".

During training, the network adjusts its weights and biases to minimize the difference between its predicted similarity scores and the true labels (Chopra et al., 2005).

The motivation for utilizing a Siamese structure to capture a complex connection between two proteins is to ensure consistency and coherence.

This approach involves simultaneously processing both proteins using a single network with shared weights, rather than employing two independent models with different sets of parameters. The use of a Siamese architecture helps maintain a unified and synchronized representation, allowing for a more accurate analysis of the relationship between the proteins (Mahapatra & Sahu, 2021).

2.5 Architecture of the Deep Learning Model

The implemented framework utilizes a Siamese CNN architecture, which comprises two identical subnetworks sharing the same structure, variables, and weights.

Applied Siamese CNN framework is shown in Figure 2.2.

The framework is made up of three major components: a profile module based on encoded sequence of proteins, a convolutional module, and a module of prediction. Encoded proteins introduces to the Siamese architecture as training inputs. In order to

acquire similarity or dissimilarity of the two encoded sequences, each one is fed into two subnetworks that share weights.

Subsequently, the output feature vectors from each subnetwork are subtracted and aggregated, enabling a comprehensive representation of the differences between the sequences. The resulting data is then subjected to a fully-connected operation, producing a single output that encapsulates the distinctive characteristics and dissimilarities of the sequences.

The fully-connected process encompasses multiplying the input with a weight matrix and incorporating a bias vector. Subsequently, the sigmoid function is applied to transform the resulting output into a probability that spans from 0 to 1. This probability serves as an indication of the extent of similarity or dissimilarity between the two protein sequences.

Throughout the training phase, the network undergoes updates by optimizing the binary cross-entropy loss. This loss metric quantifies the disparity between the network's predictions and the true labels, aiding in the refinement of the network's performance (Geckin & Kalayci Demir, 2022).

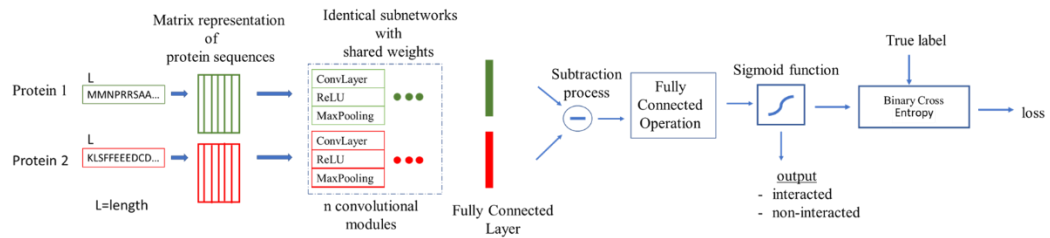


Figure 2.2 Convolutional Siamese neural network framework (Geckin & Kalayci Demir, 2022)

CHAPTER THREE

EXPERIMENTS AND RESULTS

3.1 Evaluation Metrics

To measure the performance of the implemented network, we utilize several metrics including overall prediction accuracy (ACC), specificity (SPE), precision (PRE), sensitivity (SE) or recall, Matthews Correlation Coefficient (MCC), and F-Score (F1). They are defined as follows:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \quad (3.1)$$

$$SPE = \frac{TN}{TN + FP} \quad (3.2)$$

$$PRE = \frac{TP}{TP + FP} \quad (3.3)$$

$$SE = \frac{TP}{TP + FN} \quad (3.4)$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FN) \times (TN + FP) \times (TP + FP) \times (TN + FN)}} \quad (3.5)$$

$$F - \text{Score (F1)} = \frac{2 \times (PRE \times SE)}{(PRE + SE)} \quad (3.6)$$

Where true positive (TP) is the number of interacting PPIs that are predicted correctly, true negative (TN) is the number of non interacting pairs that are correctly predicted, FP (false positive) is the number of incorrectly predicted interacting pairs, false negative (FN) is the number of interacting protein pairs that are predicted to be non interacting pairs (Chang et al., 2016).

SPE is another important metric used to assess prediction performance. It represents the ratio of TN to the total number of actual negative instances in the dataset. In the

context of PPI prediction, specificity measures the model's ability to accurately identify non-interacting protein pairs. A higher specificity value indicates a lower rate of FP, meaning that the model is effectively distinguishing non-interacting pairs.

PRE calculated as the ratio of TP to the sum of true positives and FP. PRE indicates the accuracy of the model in correctly identifying interacting protein pairs among the predicted positive instances. A higher precision value indicates a lower rate of FP, meaning that the model is making fewer incorrect predictions of interacting pairs. Precision provides insights into the reliability and quality of the positive predictions made by the model.

Recall or SE refers to the ratio of TPs to the total number of actual positive instances in the dataset. It quantifies the ability of the model to correctly identify positive instances, in this case, interacting protein pairs.

Mathews Correlation Coefficient is abbreviated as MCC, which is a correlation coefficient between the observed and predicted results. This metric is used for the measure of the quality of binary classifications. MCC equal to 1 denotes perfectly accurate prediction, -1 denotes completely incorrect prediction, and 0 denotes fully random prediction (Y. Li et al., 2021).

F-Score, also known as F1 score, is a composite metric that combines precision and recall (sensitivity) into a single value to assess the overall prediction performance of a model. It is calculated as the harmonic mean of precision and recall, giving equal weight to both metrics.

In the context of PPI prediction, the F-score represents the overall accuracy and effectiveness of the model in correctly identifying interacting protein pairs while minimizing FP and FN. A higher F-Score indicates better overall prediction performance.

3.2 The Siamese CNN Method's Performance with Binary Encoding

In this study, two distinct sets of test data were utilized to determine the performance of the Siamese CNN. Richoux's test data contains the mirror copy of each protein pairs which is also appear in the trainig set. They are called these train and test

sets as regular set. On the other hand, regular set may have a flaw: mirror couples of train proteins appear in the test set can cause to sort of overfitting problem. In order to mitigate the challenge of overfitting, a filtered dataset was curated for the study by selectively eliminating mirror copies of protein pairs from the test data that were originally present in the training data. This approach aimed to promote generalization and prevent the model from relying excessively on specific instances.

Consequently, we conducted the experiments using the filtered dataset in order to evaluate how the curated dataset affects the performance and robustness of the model. Furthermore, we employed the regular dataset to conduct a comparative analysis of the performances between Richoux et al.'s network model and our own, specifically focusing on the Binary Encoding of proteins. Table 3.1 shows the number of protein couples in regular and filtered datasets.

Table 3.1 Number of protein couples in data sets

Datasets	Number of Train Data	Number of Test Data
Regular Dataset	52,606	6574
Filtered Dataset	52,606	4068

Our study aimed to assess the performance of the recently proposed features generated using the Binary Encoding method for protein representation.

We train and test our network using 52,606 and 4068 pair of proteins, respectively. We set different hyper-paramaters to find best performance. Based on the data presented in Table 3.2, the highest accuracies of the different layer configurations of four experimental groups were 76.60%, 78.67%, 80.20%, and 77.67% for Binary Encoding method on filtered dataset. Each convolution module within the framework is composed of three layers: a 2D-convolution layer, a rectified linear unit layer, and a 2D-pooling layer. The series of convolutional layers follows the input layer. Once the final convolutional layer is specified, a FCL is added and flattened into one single vector. Various hyper-parameter configurations are employed to identify the optimal network settings. Key hyper-parameters, including batch sizes, the number of neurons in the FCL, the learning rate and the number of iterations have been fine-tuned for improved performance. In our experimental setup, we employed various learning rates,

namely 1e-03, 5e-03, 1e-04, 5e-04, 1e-05, and 6e-05. Additionally, we explored different training batch sizes, specifically 10, 16, 20, 32, 50, and 64. The neuron count in the FCL was varied within the range of 100 to 1500, while the iteration number spanned from 10,000 to 50,000.

First network consist of four 2D-convolutional and ReLU layers and three max-pooling layers. The first convolutional layer employs a 3×3 filter size and consists of 32 filters. The 'padding' parameter is set to 'same' to ensure the output feature maps have the same spatial dimensions as the input. The weights and biases of this layer are initialized using the 'narrow-normal' scheme. A ReLU layer is applied after the convolutional layer to introduce non-linearity.

Subsequently, a max pooling with a 2×2 pooling window and a stride of 2 is used to downsample the feature maps. This process aids in reducing spatial dimensions while preserving important features.

Similarly, additional convolutional layers followed by ReLU layers and max pooling layers are included in the network. Finally, a FCL is utilized with 100 neurons. This layer connects all the neurons from the previous layer and serves as a bottleneck to capture high-level representations.

The weights and biases of this FCL are initialized using the 'narrow-normal' scheme. The learning rate for the convolutional layer configuration was adjusted, ranging from 1e-03 to 6e-05.

During training, we discovered that the cross-entropy loss was very high and the learning rate was slow. Other parameters remained fixed, but the neuron count in the FCL increased by 250. However, the variations in the neuron count of FCL did not yield a very convincing result.

In the initial and second set of experiments, the filter size was maintained at a constant (3×3 , 5×5) value, while the number of filters in each convolutional layer was varied within a range of 16 to 128. The obtained best performances for these experiments were 76.60% and 78.66%, respectively. For the set of experiments the maximum values for PRE, SPE, SE, F-Score, and MCC obtained as in Table 3.2.

In the third group of experiments, non-square filters of sizes 5×3 , 7×3 , and 9×3 were employed for the four 2D convolutional layers, with filter sizes of 32, 64, and 128. The experimental results indicate that non-square filter sizes yield better performance, reaching up to 80.24% accuracy, compared to square ones.

In the last group of experiments, the objective was to evaluate the accuracy performance by employing a combination of square and non-square filters. However, the results demonstrated a decline in the maximum accuracy, which reached 77.67%.

Table 3.2 Performance of the proposed network for different hyper-parameter sets of Binary Encoding on filtered dataset

Hyper-parameters Set Number	ACC (%)	PRE (%)	SPE (%)	SE (%)	FScore (%)	MCC (%)
Set 1	76.6	85.07	87.82	65.93	74.29	55
Set 2	78.67	84.47	88.27	68.37	75.57	58
Set 3	80.2	84.89	86.5	74.04	79.1	61
Set 4	77.67	87.44	90	65.97	75.2	57

Based on the experimental findings, the optimal hyper-parameter set, denoted as in the Table 3.3 was determined.

Table 3.3 Hyper-parameters, activations, and output sizes for the Siamese CNN model for binary representation on a filtered dataset (Geckin & Kalayci Demir, 2022)

Layer	Hyper-parameters	Activations	Output Size
Input	Input Size=1166x24	1166x24	[1166 24]
1st 2D Conv. Layer	Filters=64, Filter size=[7 3] Stride=1, padding=same Activation=ReLU	1166x24x64	[1166 24]
Max-Pool	Pool size=[2 2], Stride=2, padding=[0 0 0 0]	583x12x64	[583 12]
2nd 2D Conv. Layer	Filters=64, Filter size=[7 3] Stride=1, padding=same Activation=ReLU	583x12x64	[583 12]
Max-Pool	Pool size=[2 2], Stride=2, padding=[0 0 0 0]	291x6x64	[291 6]
3rd 2D Conv. Layer	Filters=64, Filter size=[7 3] Stride=1, padding=same Activation=ReLU	291x6x64	[291 6]
Fully Connected	Activation=sigmoid	250	250

3.3 The Siamese CNN Method's Performance with AC Encoding

The AC dataset was constructed by deleting from the main dataset protein pairs containing uncommon amino acids, Z, U, B, and X, producing 26,289 interacted and 26,206 non-interacted protein pairs. For the test set filtered one already does not contain the uncommon aminoacids. So we are able to use to test our results using filtered dataset. As indicated in the Table 3.4, the test data set contains 797 protein pairs that interact and 3271 protein pairs that do not interact.

Table 3.4 Number of protein couples in dataset for AC

	Number of Train Data	Number of Test Data
Number of interacted protein pairs	26,289	797
Number of non-interacted protein pairs	26,206	3271

The lag value was set to 15, and a total of seven physicochemical properties were utilized. Consequently, the image input size was defined as 15×7 . The convolutional layer consisted of two 2D-convolutional layers with filter sizes of 3×3 , comprising 128 and 64 filters, respectively. These layers were subsequently followed by ReLU activations and 2D-max-pooling, with a pooling size and stride set to 2.

After configuring the convolutional layers, the optimization of other parameters was pursued. A series of experiments were conducted to optimize the size of the FCL. For each epoch, the network was trained using 1000 inputs, and the total number of iterations was set to 10,000. These experimental settings allowed for an extensive exploration of the parameter space and facilitated the identification of an optimal FCL size.

First, the neuron count in the FCL is varied from 100 to 2000. 100 gives the 62.2% accuracy then, the neuron count of the FCL increased to 500 and obtained significant increase (71.46%) on the accuracy. After, we tried 1000, 1200, 1400, 1500 and 2000 to obtain the maximum accuracy. As you can see from the Figure 3.1 obtained accuracy values are 77.63%, 77.44%, 78.13%, 77.04% and 77.41%.

Based on the observations, it can be concluded that the prediction performance is influenced by the number of neurons in the layers. A decrease in prediction accuracy is observed when there are fewer neurons in the network. However, as the number of neurons increases, the prediction accuracy significantly improves up to a certain point. Beyond this optimal value, adding more neurons leads to a decrease in prediction accuracy. Therefore, there exists an optimal number of neurons that yields the highest prediction accuracy, while exceeding this value decreases the network performance.

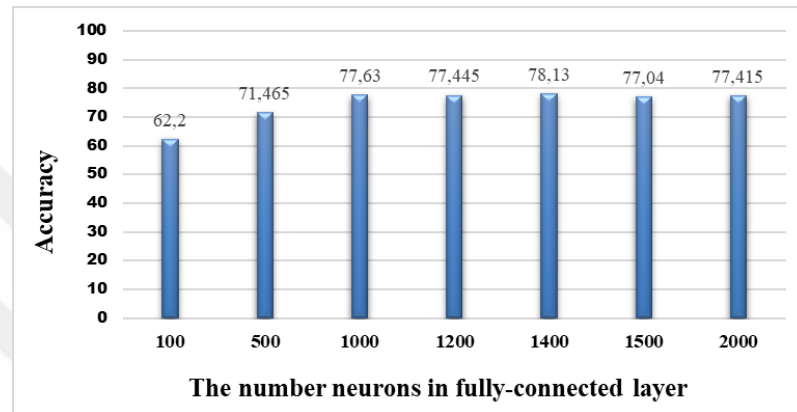


Figure 3.1 The effect of the neuron count in the FCL on the accuracy

Now FCL set as 1400, and we would like to see the effect of layers. The iteration number, learning rate, train size were kept constant at 15,000, $6e-04$ and 10,000, respectively. First, convolutional layer contains two 2D-convolutional layers with the size of 3×3 and the numbers are 128 and 64 respectively, and following with ReLU and 2D-max-pooling layer which the size and stride are set to 2. The average prediction accuracy obtained as 76.33%. Secondly, number of 2D-convolutional layer increased to obtain the effect on the accuracy. So second convolutional layer increased from 64 to 128. As a result, prediction accuracy increased up to 77.46%.

In the third experiment, another max-pooling and convolution layers added into the network. Max-pooling layer has size and stride of 2. Convolution layer includes 64 kernels with the size of 3×3 . Pooling is used to downsize the input volume and we might loosing information in max pooling layer. Due to the information loss because of the second max-pooling layer prediction accuracy decreased to 76.46%. After that, second max-pooling layer cancelled and third convolution layer increased to 128 kernels. Prediction accuracy increased to 78.60%.

As a final, number of the filters in the convolutional layers increased to 256. Obtained a slight decrease on the accuracy as 77.33%. Table 3.5 illustrates the performance of the network across various layer configurations. The highest accuracy is achieved when the network consists of three convolution layers, each with a size of 3×3 and the number of neurons 128. In the depicted network architecture, the first convolution layer is followed by a max pooling layer, which allows for spatial down-sampling. However, unlike the first layer, the second and third convolution layers do not undergo the max pooling process. This decision is made to preserve more detailed spatial information at deeper layers, enabling the network to capture finer features and patterns. This choice is motivated by the objective of capturing a greater amount of information. This particular configuration yields an accuracy of 78.60%.

Table 3.5 The effect of different layer configurations on the accuracy of AC method

	Layer Conf. 1	Layer Conf. 2	Layer Conf. 3	Layer Conf. 4	Layer Conf. 5
1st Convolution	1st convolution	1st convolution	1st convolution	1st convolution	1st convolution
	[3 3], 128	[3 3], 128	[3 3],128	[3 3],128	[3 3], 256
1st ReLU	ReLU	ReLU	ReLU	ReLU	ReLU
1st max-pooling	max-pooling	max-pooling	max-pooling	max-pooling	max-pooling
	[2 2], stride:2	[2 2], stride:2	[2 2], 'stride' 2	[2 2], 'stride' 2	[2 2], 'stride' 2
2nd Convolution	2nd convolution	2nd convolution	2nd convolution	2nd convolution	2nd convolution
	[3 3], 64	[3 3], 128	[3 3],128	[3 3], 128	[3 3], 256
2nd ReLU	ReLU	ReLU	ReLU	ReLU	ReLU
2nd max-pooling			max-pooling		
			[2 2], 'stride' 2		
3rd Convolution			3rd convolution	3rd convolution	3rd convolution
			[3 3],64	[3 3],64	[3 3],64
3rd ReLU			ReLU	ReLU	ReLU
ACC (%)	76.33	77.47	76.46	78.6	77.33
PRE (%)	87.52	85.05	82.2	85.47	84.48
SPE (%)	90.76	88.46	84.93	86.94	88.95
SE (%)	62.43	66.35	69.6	70.9	64.82
FScore (%)	72.88	74.55	75.38	77.51	73.33
MCC (%)	55	56	55	58	56

The selection of an appropriate learning rate plays a crucial role in determining the accuracy of a CNN. Unlike some other neural network architectures, CNNs do not have predefined or default learning rates. Consequently, it becomes imperative to thoroughly evaluate the influence of different learning rates on network accuracy. To achieve this, a series of experiments were conducted, systematically varying the learning rate across a range of values. By carefully analyzing the outcomes, the learning rate that produced the highest network accuracy was identified as the optimal choice.

3.4 The Siamese CNN Method's Performance with PSSM Encoding

We also conducted a comprehensive exploration of the network hyper-parameters, specifically focusing on the PSSM representations. In the PSSM model, proteins are represented by vectors of size $L \times 20$, where L denotes the length of the protein and 20 represents the twenty amino acids.

Additionally, in the PSSM encoding method, it is necessary to remove protein pairs that contain uncommon amino acids such as Z, U, B, and X from the primary dataset. Therefore, the same dataset that was prepared for the AC method was also utilized in this case. This process yielded a train dataset consisting of 26,289 interacted and 26,206 non-interacted protein pairs. Test set comprising 797 interacted and 3271 non-interacted protein pairs.

Since proteins can vary in length, it poses a challenge when inputting them into the model as they should all have the same vector size. To ensure uniformity in input dimensions, we employed zero padding on sequences, aiming to achieve the same length due to the presence of a maximum length constraint of 1166. Consequently, each pair of residues in the network input is encoded as a 1166×20 dimensional vector. This encoding scheme allows for consistent representation of the protein sequences, facilitating compatibility with the network architecture.

In the evaluation process, various kernel scales are examined for convolutional layers. Specific kernel combinations were evaluated to assess their impact on the overall performance of the network and the resulting accuracies obtained from these configurations.

In addition to exploring various kernel combinations and filter sizes, we also investigated the impact of different parameters on the performance of the model. These parameters included the neuron count in the FCL, the number of iterations, and the learning rate. In our experimental configuration, we utilized a range of learning rates, specifically $1e-03$, $5e-03$, $1e-04$, $5e-04$, $1e-05$, and $6e-05$. Additionally, we investigated different training batch sizes, namely 10, 16, 20, 32, 50, and 64. The neuron count in the FCL was varied between 100 and 1500, while the iteration number ranged from 10,000 to 50,000.

The first group consists of four convolutional layers. The first and second convolutions utilize a filter size of 7×7 , with 128 and 64 filters, respectively. The third and fourth convolutions employ a filter size of 5×3 , with 64 filters each.

Additionally, two max-pooling layers are incorporated in the architecture, which allow for downsampling of the feature maps.

ReLU activation functions are applied after each convolutional layer, totaling four ReLU activations within the seven-layer group.

This configuration of convolutional layers, max-pooling layers, and ReLU activations is employed to assess the impact of different kernel scales on the network's performance. The learning rate is set to $6e-05$. When configuring the FCL, we experimented with different numbers of neurons, specifically 1200, 1300, and 1400. The Figure 3.2 illustrates the corresponding mean accuracies obtained from the different configurations of the FCL. The accuracy values are as follows: 83.99%, 82.73%, and 83.64%. These results indicate the performance of the network in terms of prediction accuracy when varying the neuron count in the FCL.

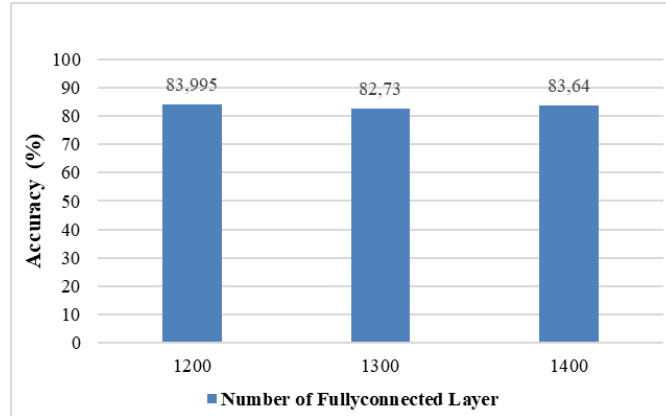


Figure 3.2 Effect of the neuron count in the FCL

The utilization of 7×7 filter sizes in convolutional layers is intended to capture large-scale features present in the input data. These layers are typically positioned at the initial stages of the Siamese CNN architecture, aiming to detect and extract low-level features, such as edges, corners, and blobs. The larger receptive field offered by the 7×7 filters enables the network to analyze a broader context and identify spatial relationships within the data. On the other hand, the incorporation of 5×3 filter sizes serves the purpose of capturing smaller-scale features. These layers are commonly situated deeper within the Siamese CNN architecture, as they are designed to capture higher-level features, including object parts, texture variations, and intricate patterns. The smaller receptive field of the 5×3 filters allows for more detailed analysis of localized features, enabling the network to discern finer nuances within the input data.

In the first group, four convolutional layers were utilized. The first and second convolutions had a filter size of 7×3 , with 128 and 64 filters respectively. The third and fourth convolutions had a filter size of 5×3 , with 64 filters each. The architecture also included two max-pooling layers and four ReLU layers.

The learning rate was set to $6e-05$, and the accuracy obtained as 83.00% when the number of neurons in the FCL was set to 1400.

This configuration allowed the network to capture both larger-scale and smaller-scale features present in the input data. The introduction of different filter sizes in the convolutional layers facilitated the extraction of various levels of visual information, contributing to the improvement in accuracy. Additionally, adjusting the neuron count

in the FCL further optimized the model's performance, resulting in the observed increase in accuracy.

In second group of convolutional layers, we investigated the effect of using smaller 3×3 filter sizes on the accuracy. This group consists of three convolutional layers, each serving a specific purpose in feature extraction.

The first and second convolutions employ a filter size of 3×3 , with 128 and 64 filters respectively. These layers are designed to learn higher-level features by combining the low-level features learned by the previous group of convolutional layers. The higher number of filters in the first convolutional layer, compared to the second one, suggests that more complex and higher-level features are expected to be learned in the initial layer.

On the other hand, the third convolution also utilize a filter size of 3×3 , but with 64 filters each. The objective of this layer is to further capture higher-level features while reducing the risk of overfitting by employing fewer filters. Additionally, it contribute to the reduction of dimensionality by decreasing the number of output feature maps.

Table 3.6 Performance of the proposed network for different hyper-paramater sets of AC on filtered dataset

Hyper-parameters Set Number	ACC (%)	PRE (%)	SPE (%)	SE (%)	F-Score (%)	MCC (%)
Set 1	83	89.11	90.34	75.95	82	67
Set 2	83.73	90.39	92.08	73.51	81.08	67
Set 3	84.07	91.45	92.45	76.17	83.11	69
Set 4	81.93	87.05	89.21	74.46	80.26	64

In the third group of the experimental series, our investigation focuses on examining the impact of combining square and non-square filters on network performance. We explore various combinations, such as 3×3 and 5×3 , 3×3 and 7×3 , as well as 5×5 and 7×3 filters, utilizing optimal parameters. The highest achieved accuracy in this experimentation was recorded at 81.93%. By systematically adjusting hyper-parameters and evaluating the resulting accuracies, we identified the optimal parameter settings that led to a significant increase in accuracy, reaching 84.07%.

Overall, by incorporating smaller filter sizes in the third group of convolutional layers, the network aims to extract and learn more intricate and sophisticated features, leading to improved accuracy (84.07%).

The optimal parameter settings that resulted in an increased accuracy of 84.07% were as follows: the neuron count in the FCL set to 1400, an iteration number of 50,000, and a learning rate of 1e-05. These specific values and configurations were carefully selected and fine-tuned to maximize the model's accuracy through experimentation and analysis. The accuracy improvement achieved with these optimized parameter settings is depicted in Table 3.7.

Table 3.7 The effect of different configurations on the accuracy of PSSM

ACC (%)	84.06
Layer configuration Kernel size Number of the kernels	1st convolution [3 3],128
	ReLU
	max-pooling [2 2], 'stride' 2
	2nd convolution [3 3],128
	ReLU
	max-pooling [2 2], 'stride' 2
	3rd convolution [3 3], 64
Number of neurons in FCL	1400
Learning rate	1,00E-05
Number of iterations	50

By strategically placing convolutional layers with different filter sizes at various depths in the Siamese CNN architecture, the network can effectively capture both large-scale and small-scale features, enabling comprehensive feature extraction and enhancing its ability to understand and interpret complex visual information.

3.5 Performance Comparison of Three Protein Encoding Method

We investigated the utilized network performance using three different protein encoding methods: Binary Encoding, AC, and PSSM.

From Table 3.8, we can obtain that the PSSM protein encoding method gives good results with the average prediction ACC, PRE, SPE, SE, F-Score and MCC of 84.07%, 91.45%, 92.45%, 76.17%, 83.11%, and 69%, respectively, using the regular dataset. Binary Encoding obtained an average prediction ACC, PRE, SPE, SE, F-Score and

MCC of 80.2%, 84.86%, 86.5%, 74.04%, 79.1%, and 61%, respectively, using the regular dataset. AC method obtained an average prediction ACC, PRE, SPE, SE, F-Score and MCC of 78.6%, 85.47%, 86.94%, 70.09%, 77.51%, and 58%, respectively, using the regular dataset.

The network yielded superior results when utilizing the PSSM encoding method. This outcome can potentially be attributed to the fact that PSSMs incorporate information from the evolutionary history of proteins, which has been demonstrated to possess greater predictive power for PPIs compared to the other two sequence-based protein encoding methods.

The Binary Encoding method also demonstrated promising results in terms of prediction accuracy. One notable advantage of this method is its faster protein encoding compared to the AC and PSSM methods. This efficiency in encoding enables quicker processing and analysis of protein sequences, making it advantageous in scenarios where computational speed is a critical factor. Indeed, Binary Encoding based on the one-hot encoding method results in multi-dimensional and sparse vector representations. These characteristics can introduce complexity to machine learning models. The multi-dimensionality of the input data increases computational requirements and can potentially lead to overfitting if not carefully managed. Additionally, the sparse nature of the data can pose challenges in terms of memory usage and computational efficiency. Therefore, when working with Binary Encoding, it is crucial to employ appropriate techniques for dimensionality reduction, feature selection, or regularization to mitigate these complexities and improve the performance of machine learning models.

AC method takes the neighbouring effect into account. The results indicate that the AC method has slightly lower accuracy compared to both the PSSM and Binary Encoding methods in the context of the regular dataset. Conversely, one significant challenge in current research is the slow speed of learning algorithms for PPI prediction. However, AC method exhibits faster training speeds compared to the other two protein sequence encoding methods. This advantage in training speed can be advantageous in reducing the computational time required for training models and accelerating the overall PPI prediction process (F. Li et al., 2020).

Table 3.8 Performance comparison of three encoding methods

Protein Encoding Method	ACC (%)	PRE (%)	SPE (%)	SE (%)	F-Score (%)	MCC (%)
Binary Encoding	80.2	84.89	86.5	74.04	79.1	61
AC	78.6	85.47	86.94	70.9	77.51	58
PSSM	84.07	91.45	92.45	76.17	83.11	69

3.6 Comparative Analysis of the Prediction Performances

To ensure a comprehensive evaluation of the proposed method, we conducted a comparative analysis by comparing its results with Richoux et al.'s work. The performance evaluation of our network involved utilizing the Richoux et al.'s regular dataset.

The experiments were divided into four groups, each aimed at finding the optimal layer configuration using square and non-square filters. The achieved accuracies for each configuration can be observed in Table 3.9.

Table 3.9 Performance comparison of different hyper-parameters on regular dataset using binary encoding

Testing Hyper-parameters	ACC (%)
Set 1	79.88
Set 2	85.36
Set 3	83.15
Set 4	88.87

In the first group of experiments, square filters of sizes 3×3 and 5×5 were applied to four 2D convolutional layers, with filter sizes of 32, 64, and 128. The mean accuracy for this group was calculated as 79.88%.

In the second group, non-square filters of sizes 5×3 , 7×3 , and 9×3 were employed for the four 2D convolutional layers, with the same filter sizes. Notably, this led to a significant increase in accuracy, from 79.88% to 85.36%.

The third group of experiments incorporated additional 2D convolutions, resulting in an average accuracy of 83.15%.

Finally, in the fourth group of experiments, a combination of square and non-square filters was employed. The neuron count in the FCL was varied within a range of 10 to 3000. The best performance was achieved when setting the learning rate to 6e-04. The network architecture included four 2D convolutional layers followed by ReLU and max pooling layers. The network accuracy reached 88.87%.

Table 3.10 provides a comprehensive overview of the implemented convolutional Siamese neural network, presenting the optimal hyper-parameter values, activations, and output size. This table serves as a valuable reference for analyzing the model's configuration and conducting.

According to the input size of 1166×24 non-square filters fit to the input data size and square filters captured informations. Using both square and non-square filters in one network boost accuracy by 11.25%.

Table 3.10 Siamese CNN model layer configuration, hyper-parameters, and output sizes for binary encoding on regular dataset

Layer	Hyper-parameters	Activations	Output Size
Input	Input Size=1166x24	1166×24	[1166 24]
1st Conv.	Filters=64, Filter size=[5 3] Stride=1, padding=same Activation=relu	1166×24×64	[1166 24]
Max-Pool	Pool size=[2 2], Stride=2, padding=[0 0 0 0]	583×12×64	[583 12]
2nd Conv.	Filters=64, Filter size=[5 3] Stride=1, padding=same Activation=relu	583×12×64	[583 12]
Max-Pool	Pool size=[2 2], Stride=2, padding=[0 0 0 0]	291×6×64	[291 6]
3rd Conv.	Filters=32, Filter size=[3 3] Stride=1, padding=same Activation=relu	291×6×32	[291 6]
Max-Pool	Pool size=[2 2], Stride=2, padding=[0 0 0 0]	145×3×32	[145 3]
4th Conv.	Filters=32, Filter size=[3 3] Stride=1, padding=same Activation=relu	145×3×32	[145 3]
Fully Connected	Activation=sigmoid	1080	1080

Richoux et al., utilize a fully connected deep learning model with binary encoded protein pairs to predict PPIs in their regular and strict datasets, and the results have proven its ability. Specifically, we use the same feature extraction method (Binary Encoding) to compare the network performance in the same dataset. The comparison

of network performances based regular dataset can be obtained from the Table 3.11. As can be seen from Table 3.11, when using convolutional Siamese neural network to predict PPIs of regular dataset, we perceived promising results with the ACC, PRE, SE, and F-Score of 88.87%, 90.63%, 86.63%, and 88.59%, respectively.

Our network model exhibited a higher sensitivity, with an improvement of 0.81% compared to the sensitivity of the Richoux et al.'s model. This indicates that our model was more effective in correctly identifying positive instances or interacted protein pairs.

Table 3.11 Comparison between our network and Richoux et al.'s fully connected model on regular dataset

Evaluation Metrics	Richoux's Results	Our Network
ACC (%)	89.97	88.87
PRE (%)	94.92	90.63
SE (%)	85.93	86.63
F-Score (%)	90.21	88.59

We also conducted a comparative analysis between our convolutional Siamese neural network model and Richoux's fully connected model on the strict dataset, notable differences in prediction performance metrics were observed.

The primary distinction between the strict and regular datasets lies in the composition of protein pairs. In the regular dataset, while none of the protein pairs appear in the training set, many individual proteins within the regular dataset do appear in the training set. To address the potential issue of overfitting, where the model could identify patterns like "protein X never interacts," a strict dataset was also created.

The strict dataset consists of pairs of proteins where each protein appears at most twice in the entire dataset. By enforcing this constraint, the strict dataset promotes a more balanced representation of proteins, reducing the risk of the model overly relying on specific individual proteins.

The extracted couples from the dataset were utilized to create the test set, which consists of 460 samples. Among these samples, there are 230 positive instances

(indicating interacting protein pairs) and 230 negative instances (representing non-interacting protein pairs).

Finally, the mirror copies of each pair of proteins were added to both our normal and strict sets. This modification does not exactly double the sets because the original dataset includes several mirror pairs. Table 3.12 shows the number of interacted and non-interacted protein combinations for strict test set.

Table 3.12 The number of interacted and non interacted protein pairs of strict test set

	Number of interacted protein pairs	Number of non-interacted protein pairs
Richoux et al.'s strict data set	260	460

Table 3.13 presents a comparison between our convolutional Siamese neural network model on the Richoux's strict data set. Richoux's model achieved an ACC of 78.33%, PRE of 55.76%, SE (recall) of 77.95%, and an F-score of 65.02%. In contrast, our convolutional Siamese neural network model demonstrated improved performance with an ACC of 83.6%, PRE of 98.26%, SE of 67.87%, and an F-score of 80.28%. These results indicate that our model outperformed Richoux's model in terms of ACC, PRE, and F-score. Although Richoux's model had a higher sensitivity, our model demonstrated higher overall performance across multiple metrics.

The observed outcomes suggest that our network successfully addressed the overfitting problem commonly encountered in training networks, resulting in superior performance, particularly on the strict dataset. The improved ACC, PRE, SE, and F-score of our model demonstrate its effectiveness in predicting PPIs and its capability to generalize well to unseen data.

Thus, we can conclude that our network exhibits better performance and provides an enhanced solution for the prediction of PPIs, especially when considering the prevention of overfitting and the utilization of the strict dataset.

Table 3.13 Comparison between our network and Richoux et al.'s fully connected model on strict dataset

Evaluation Metrics	Richoux's Results	Our Network
ACC (%)	78.33	83.36
PRE (%)	55.76	98.26
SE (%)	77.95	67.87
F-score (%)	65.02	80.28

3.7 Ensemble

Position-dependent methods have the capability to capture homologous information, which is valuable in understanding the evolutionary relationships between proteins. On the other hand, position-independent methods can reflect the fundamental properties of amino acids, providing insight into their inherent characteristics. These two types of methods, in a sense, can complement each other (Salinas & Ranganathan, 2018).

By combining their strengths, a more comprehensive understanding of protein properties and interactions can be achieved. Besides, different protein encoding methods can provide distinct sets of features that capture various aspects of interacting protein sequences. These encoding methods offer complementary information, enabling a more comprehensive understanding of PPIs. Therefore, it is recommended to employ multiple encoding methods instead of relying only on a single approach when predicting PPIs (Zhang et al., 2019).

In our study, we have utilized three encoding methods, namely Binary Encoding, AC, and PSSM, to extract feature information from amino acid sequences. By applying these three encoding methods, we aim to harness a broader range of information for PPI prediction. Each encoding method has its own strengths in capturing different aspects of the protein interaction network.

On this basis, a deep ensemble learning method is introduced for predicting protein interactions, offering a suitable approach for comprehensive learning while considering the benefits of various methods. In this method, protein sequences are

encoded using Binary Encoding, AC, and PSSM representations, capturing the unique characteristics of each protein within the protein interaction network. Moreover, the convolutional Siamese neural networks are employed to automatically extract protein predictions, which are subsequently integrated as new features within a multilayer feed-forward neural network.

The weights, biases, and parameter values of the network that exhibited superior prediction accuracy have been saved. Subsequently, the prediction results, represented as float values ranging between 0 and 1, were obtained and assigned to respective columns in a matrix. The first column corresponds to binary prediction values, the second column represents AC prediction values, and the third column represents PSSM prediction values for 50,000 protein pairs. Therefore, our matrix size is $50,000 \times 3$, which serves as the input for the feed-forward neural network structure. In addition to the prediction results, the corresponding labels have also been saved. These labels are associated with the 50,000 protein pairs and are stored as a matrix of size $50,000 \times 1$. The same technique is utilized to derive prediction values that likewise fall within the range of floating-point numbers between 0 and 1 on filtered dataset.

To establish the desired activation functions within the network, the transfer function settings for the respective layers are specified. The first layer, representing the hidden layer, adopts a 'hardlim' transfer function, which is a hard-limiter activation function. The second layer, constituting the output layer, employs a 'purelin' transfer function, corresponding to a linear activation function.

We have combined Siamese neural network and multilayer feed-forward neural network (MLF-NN) to increase the prediction performance of PPIs from primary sequences. The proposed model framework is shown in Figure 3.3.

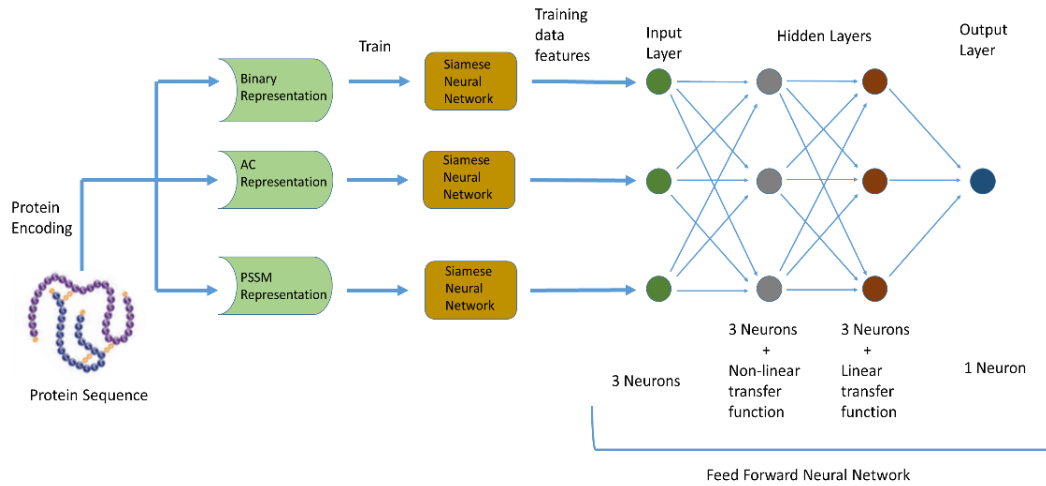


Figure 3.3 The framework for the proposed ensemble model

The employed MLF-NN structure comprises of a single hidden layer with a range of number of neurons from 2 to 16. To train the network, various backpropagation algorithms were utilized, including Levenberg-Marquardt, Bayesian Regularization, Resilient Backpropagation, Gradient Descent, Variable Learning Rate Gradient Descent, Gradient Descent with Momentum, and Fletcher-Powell Conjugate Gradient. The highest accuracy results were achieved by employing three specific training algorithms: Levenberg-Marquardt, Gradient Descent with Momentum, and Fletcher-Powell Conjugate Gradient. These algorithms demonstrated superior performance in achieving optimal accuracy levels during the training process. The performance of the network with different hidden layer configurations and transfer functions can be observed in Figure 3.4.

When using the Fletcher-Powell Conjugate Gradient backpropagation algorithm, the highest accuracy was achieved when the number of neurons in the hidden layer was set to 6. Additionally, another advantage of this algorithm is its faster training capability. This is due to the fact that the basic backpropagation algorithm adjusts weights in the steepest descent direction (negative gradient), while conjugate gradient algorithms conduct searches along conjugate directions. This approach typically leads to faster convergence compared to steepest descent directions, resulting in accelerated training of the algorithm (Andrei, 2009).

The developed ensemble model exhibited notable performance, surpassing the individual results of the encoding methods. Table 3.14 presents the performance comparison of individual encoding methods and the ensemble method. Particularly, the PSSM method yielded the highest accuracy among the encoding techniques employed. The prediction accuracy experienced a notable improvement, rising from 84.07% to 86.27%. This enhancement signifies the effectiveness and superiority of the ensemble model in accurately predicting protein interactions.

Table 3.14 Performance comparison of individual encoding methods using proposed convolutional Siamese Network and ensemble method combining Siamese and MLF-NNs on filtered dataset.

Protein Encoding Method	ACC (%)	PRE (%)	SPE (%)	SE (%)	F-Score (%)	MCC (%)
Binary Encoding	80.2	84.89	86.5	74.04	79.1	61
AC	78.6	85.47	86.94	70.9	77.51	58
PSSM	84.07	91.45	92.45	76.17	83.11	69
Ensemble Method	86.27	89.16	90.03	82.49	85.69	73

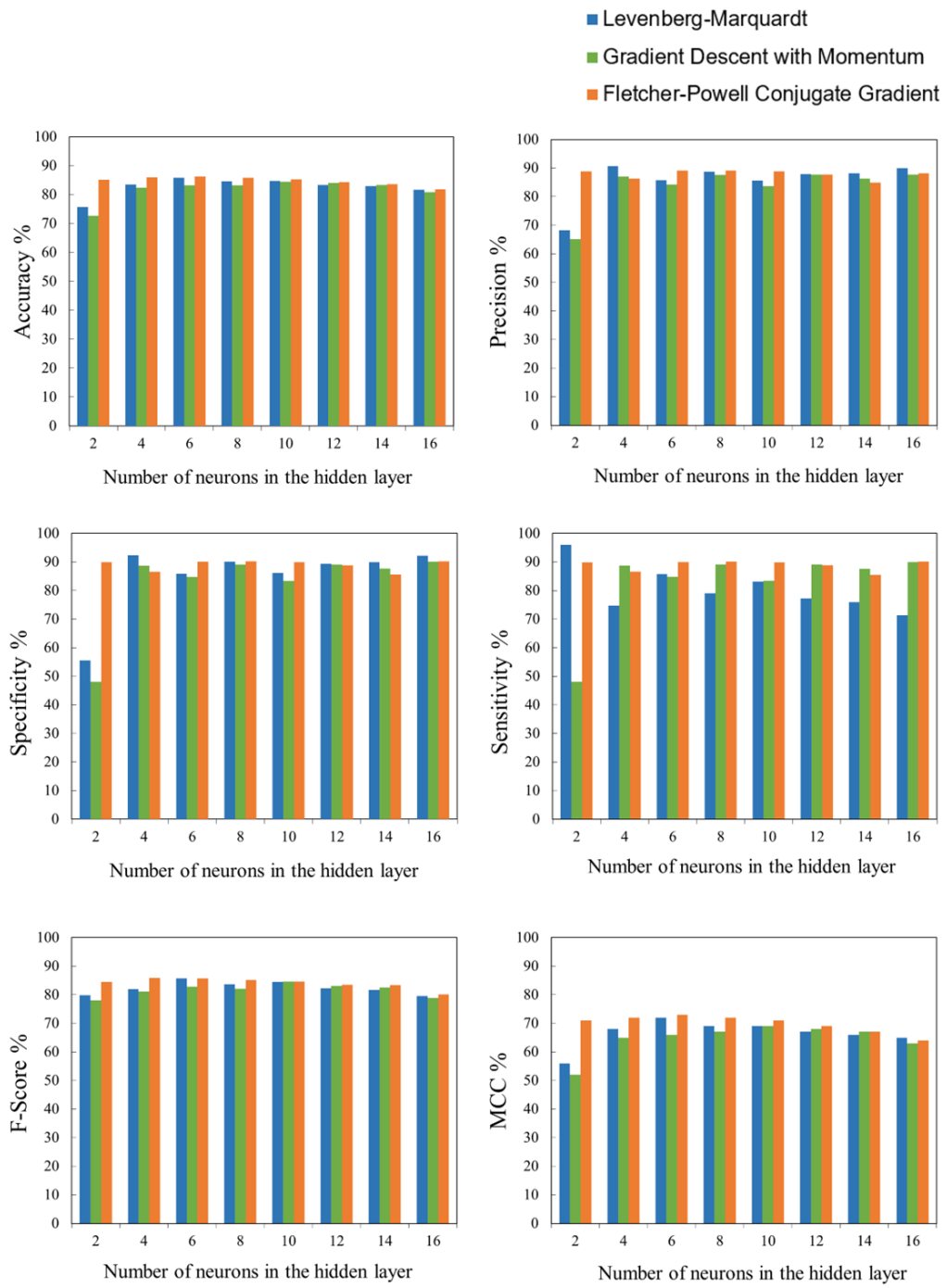


Figure 3.4 Ensemble network results with the different number of hidden neurons

CHAPTER FOUR

CONCLUSION

In this research, a convolutional Siamese neural network model was developed and utilized to predict protein-protein interactions (PPIs) only based on the sequencing data of proteins. One of the primary computational challenges in sequence-based PPI prediction methods is to identify an appropriate feature representation that effectively captures the essential information of protein interactions. To overcome this problem, we have undertaken the different encoding methods of protein sequences as an initial step in addressing this issue: Binary Encoding, AC, and PSSM methods. By comparing the prediction performances of these three encoding methods, we aimed to determine the most suitable and effective encoding method for PPI prediction. Based on the experimental results, the PSSM model exhibited superior performance in predicting PPIs compared to the AC and Binary encoding models.

The PSSM model achieved an ACC of 84.07%, PRE of 91.45%, SPE of 92.45%, SE of 76.17%, F-Score of 83.11%, and MCC of 69%. These metrics indicate the superior performance and effectiveness of the PSSM model in accurately predicting PPIs compared to the other encoding models. The use of Position Specific Scoring Matrices (PSSMs) in this study proves to be effective in preserving the evolutionary information embedded within protein sequences. By utilizing this method, the network may have been possible to capture valuable features that contribute to higher performance in PPI prediction. The inclusion of PSSMs allows the network to benefit from the informative evolutionary patterns encoded in the sequences, ultimately enhancing the prediction accuracy and overall performance of the model.

Since different encoding methodologies capture diverse features from the same protein sequence, an ensemble method was proposed to combine the outputs of these encoders, thereby improving prediction performance. The integration of multiple encoding strategies acquire complementary strengths of three methods and enhance the accuracy of PPI prediction.

On this basis, the proposed ensemble model significantly improves the prediction accuracy, achieving the ACC level of 86.27%. PRE, SPE, SE, F-Score, and MCC values are calculated, 89.16%, 90.03%, 82.49%, 85.69%, and 73% respectively.

In order to highlight the advantage of the proposed model, it is compared with Richoux et. al.'s PPI prediction study on regular and strict datasets. Prediction accuracy increase to 88.87% using the proposed convolutional Siamese network model on the regular test set. The reason of this including the mirror couples of proteins in the test set can cause to the overfitting problem. It can be seen from Table 3.11 that the comparison of network performances of two models.

The performance of the network was further evaluated by comparing its results on a strict test set, which consists of protein pairs that appear at most twice in the entire dataset. Table 3.13 clearly demonstrates that our proposed model outperforms Richoux et al.'s network in terms of prediction performance. Specifically, our model achieves an ACC of 83.36%, PRE of 98.26%, SE of 67.87%, and F-Score of 80.28%.

This experimental results show that the proposed convolutional Siamese neural network model in combination with PSSM encoding, achieved considerably improved precision in the identification of PPIs. Additionally, in this study, we have introduced an ensemble network that combines proposed deep learning method with a multilayer feed-forward neural network for PPI prediction. The utilization of this ensemble method further improves the prediction performance.

REFERENCES

- Akram, M., Asif, M., Uzair, M., Naveed, A., Madni, M. A., Ali Shah, D. S., Hasan, Z., & Khan, A. (2011). Amino acids: A review article. *Journal of Medicinal Plants Research*, 5, 3997–4000.
- Alain, G., Bengio, Y., Courville, A., Fergus, R., & Manning, C. (2014). What Regularized Auto-Encoders Learn from the Data-Generating Distribution. In *Journal of Machine Learning Research* (Vol. 15).
- Alonso-López, Di., Campos-Laborie, F. J., Gutiérrez, M. A., Lambourne, L., Calderwood, M. A., Vidal, M., & De Las Rivas, J. (2019). APID database: Redefining protein-protein interaction experimental evidences and binary interactomes. *Database*, 2019. <https://doi.org/10.1093/database/baz005>
- Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., & Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. In *Nucleic Acids Research* (Vol. 25, Issue 17). Oxford University Press.
- An, J. Y., Zhou, Y., Zhao, Y. J., & Yan, Z. J. (2019). An Efficient Feature Extraction Technique Based on Local Coding PSSM and Multifeatures Fusion for Predicting Protein-Protein Interactions. *Evolutionary Bioinformatics*, 15. <https://doi.org/10.1177/1176934319879920>
- Andrei, N. (2009). Acceleration of conjugate gradient algorithms for unconstrained optimization. *Applied Mathematics and Computation*, 213, 361–369. <https://doi.org/10.1016/j.amc.2009.03.020>
- Bader, G. D., Donaldson, I., Wolting, C., Ouellette, B. F. F., Pawson, T., & Hogue, C. W. V. (2001). BIND-The Biomolecular Interaction Network Database. In *Nucleic Acids Research* (Vol. 29, Issue 1). <http://www.fsf.org/>
- BROMLEY, J., BENTZ, J. W., BOTTOU, L., GUYON, I., LECUN, Y., MOORE, C., SÄCKINGER, E., & SHAH, R. (1993). Signature Verification Using a “Siamese” Time Delay Neural Network. *International Journal of Pattern*

Recognition and Artificial Intelligence, 07(04), 669–688.
<https://doi.org/10.1142/s0218001493000339>

Chang, J. W., Zhou, Y. Q., Ul Qamar, M. T., Chen, L. L., & Ding, Y. D. (2016). Prediction of protein–protein interactions by evidence combining methods. In *International Journal of Molecular Sciences* (Vol. 17, Issue 11). MDPI AG.
<https://doi.org/10.3390/ijms17111946>

Chen, J. Y., Mamidipalli, S. R., & Huan, T. (2009). HAPPI: An online database of comprehensive human annotated and predicted protein interactions. *BMC Genomics*, 10(SUPPL. 1). <https://doi.org/10.1186/1471-2164-10-S1-S16>

Chen, M., Ju, C. J. T., Zhou, G., Chen, X., Zhang, T., Chang, K. W., Zaniolo, C., & Wang, W. (2019). Multifaceted protein-protein interaction prediction based on Siamese residual RCNN. *Bioinformatics*, 35(14), i305–i314.
<https://doi.org/10.1093/bioinformatics/btz328>

Chen, W., Wang, S., Song, T., Li, X., Han, P., & Gao, C. (2022). DCSE:Double-Channel-Siamese-Ensemble model for protein protein interaction prediction. *BMC Genomics*, 23(1). <https://doi.org/10.1186/s12864-022-08772-6>

Chopra, S., Hadsell, R., & Lecun, Y. (2005). *Learning a Similarity Metric Discriminatively, with Application to Face Verification*.

Chou, K. C. (2005). Using amphiphilic pseudo amino acid composition to predict enzyme subfamily classes. *Bioinformatics*, 21(1), 10–19.
<https://doi.org/10.1093/bioinformatics/bth466>

Chung, C. R., Jhong, J. H., Wang, Z., Chen, S., Wan, Y., Horng, J. T., & Lee, T. Y. (2020). Characterization and identification of natural antimicrobial peptides on different organisms. *International Journal of Molecular Sciences*, 21(3).
<https://doi.org/10.3390/ijms21030986>

Durham, E., Dorr, B., Woetzel, N., Staritzbichler, R., & Meiler, J. (2009). Solvent accessible surface area approximations for rapid and accurate protein structure

- prediction. *Journal of Molecular Modeling*, 15(9), 1093–1108.
<https://doi.org/10.1007/s00894-009-0454-9>
- Fauchere', J.-L., Charton', M., Kier', L. B., Verloop4, A., & Pliska', V. (1988). Amino acid side chain parameters for correlation studies in biology and pharmacology. In *Int. J. Peptide Protein Res* (Vol. 32).
- Gao, Z. G., Wang, L., Xia, S. X., You, Z. H., Yan, X., & Zhou, Y. (2016). Ens-PPI: A Novel Ensemble Classifier for Predicting the Interactions of Proteins Using Autocovariance Transformation from PSSM. *BioMed Research International*, 2016. <https://doi.org/10.1155/2016/4563524>
- Geckin, D., & Kalayci Demir, G. (2022). Sequence Based Prediction of Protein-Protein Interactions via Siamese Neural Networks. *TIPTEKNO 2022 - Medical Technologies Congress, Proceedings*.
<https://doi.org/10.1109/TIPTEKNO56568.2022.9960159>
- Gribskov, M., Mclachlant, A. D., & Eisenberg, D. (1987). Profile analysis: Detection of distantly related proteins. In *Biochemistry* (Vol. 84).
- Guo, Y., Yu, L., Wen, Z., & Li, M. (2008). Using support vector machine combined with auto covariance to predict protein-protein interactions from protein sequences. *Nucleic Acids Research*, 36(9), 3025–3030.
<https://doi.org/10.1093/nar/gkn159>
- Hashemifar, S., Neyshabur, B., Khan, A. A., & Xu, J. (2018). Predicting protein-protein interactions through sequence-based deep learning. *Bioinformatics*, 34(17), i802–i810. <https://doi.org/10.1093/bioinformatics/bty573>
- Hendy, H., Khalifa, W., Roushdy, M., & Salem, A. B. (2015). A STUDY OF INTELLIGENT TECHNIQUES FOR PROTEIN SECONDARY STRUCTURE PREDICTION. In *International Journal "Information Models and Analyses* (Vol. 4, Issue 1).

- Hess, M., Keul, F., Goesele, M., & Hamacher, K. (2016). Addressing inaccuracies in BLOSUM computation improves homology search performance. *BMC Bioinformatics*, 17(1). <https://doi.org/10.1186/s12859-016-1060-3>
- Hopp, T. P., & Woods, K. R. (1981). *Prediction of protein antigenic determinants from amino acid sequences (hydrophilicity analysis/protein conformation)* (Vol. 78, Issue 6).
- Imai, T., Kinoshita, M., & Hirata, F. (2000). Theoretical study for partial molar volume of amino acids in aqueous solution: implication of ideal fluctuation volume. *Journal of Chemical Physics*, 112(21), 9469–9478. <https://doi.org/10.1063/1.481565>
- Ingber, D. E. (2000). The origin of cellular life. In *BioEssays* (Vol. 22). John Wiley & Sons, Inc.
- Jha, K., & Saha, S. (2020). Amalgamation of 3D structure and sequence information for protein–protein interaction prediction. *Scientific Reports*, 10(1). <https://doi.org/10.1038/s41598-020-75467-x>
- Jha, K., Saha, S., & Singh, H. (2022). Prediction of protein–protein interaction using graph neural networks. *Scientific Reports*, 12(1). <https://doi.org/10.1038/s41598-022-12201-9>
- Jing, X., Dong, Q., Hong, D., & Lu, R. (2020). Amino Acid Encoding Methods for Protein Sequences: A Comprehensive Review and Assessment. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 17(6), 1918–1931. <https://doi.org/10.1109/TCBB.2019.2911677>
- Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S. A. A., Ballard, A. J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., ... Hassabis, D. (2021). Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873), 583–589. <https://doi.org/10.1038/s41586-021-03819-2>

- Kerrien, S., Aranda, B., Breuza, L., Bridge, A., Broackes-Carter, F., Chen, C., Duesbury, M., Dumousseau, M., Feuermann, M., Hinz, U., Jandrasits, C., Jimenez, R. C., Khadake, J., Mahadevan, U., Masson, P., Pedruzzi, I., Pfeifferberger, E., Porras, P., Raghunath, A., ... Hermjakob, H. (2012). The IntAct molecular interaction database in 2012. *Nucleic Acids Research*, 40(D1). <https://doi.org/10.1093/nar/gkr1088>
- Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Neural Information Processing Systems*, 25. <https://doi.org/10.1145/3065386>
- Kuzmanov, U., & Emili, A. (2013). Protein-protein interaction networks: Probing disease mechanisms using model systems. In *Genome Medicine* (Vol. 5, Issue 4). <https://doi.org/10.1186/gm441>
- Lamesch, P., Berardini, T. Z., Li, D., Swarbreck, D., Wilks, C., Sasidharan, R., Muller, R., Dreher, K., Alexander, D. L., Garcia-Hernandez, M., Karthikeyan, A. S., Lee, C. H., Nelson, W. D., Ploetz, L., Singh, S., Wensel, A., & Huala, E. (2012). The Arabidopsis Information Resource (TAIR): Improved gene annotation and new tools. *Nucleic Acids Research*, 40(D1). <https://doi.org/10.1093/nar/gkr1090>
- Lee, W. Y., Park, S. M., & Sim, K. B. (2018). Optimal hyperparameter tuning of convolutional neural networks based on the parameter-setting-free harmony search algorithm. *Optik*, 172, 359–367. <https://doi.org/10.1016/j.ijleo.2018.07.044>
- Li, F., Zhu, F., Ling, X., & Liu, Q. (2020). Protein Interaction Network Reconstruction Through Ensemble Deep Learning With Attention Mechanism. *Frontiers in Bioengineering and Biotechnology*, 8. <https://doi.org/10.3389/fbioe.2020.00390>
- Li, H., Gong, X. J., Yu, H., & Zhou, C. (2018). Deep neural network based predictions of protein interactions using primary sequences. *Molecules*, 23(8). <https://doi.org/10.3390/molecules23081923>

- Li, J., & Chen, Y. (2013). Auto Covariance Combined with Artificial Neural Network for Predicting Protein-Protein Interactions. *Advanced Materials Research*, 765–767. <https://doi.org/10.2991/icsem.2013.153>
- Li, Y., Li, L. P., Wang, L., Yu, C. Q., Wang, Z., & You, Z. H. (2019). An ensemble classifier to predict protein–protein interactions by combining PSSM-based evolutionary information with local binary pattern model. *International Journal of Molecular Sciences*, 20(14). <https://doi.org/10.3390/ijms20143511>
- Li, Y., Wang, Z., Li, L. P., You, Z. H., Huang, W. Z., Zhan, X. K., & Wang, Y. Bin. (2021). Robust and accurate prediction of protein–protein interactions by exploiting evolutionary information. *Scientific Reports*, 11(1). <https://doi.org/10.1038/s41598-021-96265-z>
- Li, Y., Wang, Z., You, Z. H., Li, L. P., & Hu, X. (2022). Predicting Protein-Protein Interactions via Random Ferns with Evolutionary Matrix Representation. *Computational and Mathematical Methods in Medicine*, 2022. <https://doi.org/10.1155/2022/7191684>
- Licata, L., Briganti, L., Peluso, D., Perfetto, L., Iannuccelli, M., Galeota, E., Sacco, F., Palma, A., Nardozza, A. P., Santonico, E., Castagnoli, L., & Cesareni, G. (2012). MINT, the molecular interaction database: 2012 Update. *Nucleic Acids Research*, 40(D1). <https://doi.org/10.1093/nar/gkr930>
- Lin, K., May, A. C. W., & Taylor, W. R. (2002). Amino acid encoding schemes from protein structure alignments: Multi-dimensional vectors to describe residue types. *Journal of Theoretical Biology*, 216(3), 361–365. <https://doi.org/10.1006/jtbi.2001.2512>
- Lin, N., Wu, B., Jansen, R., Gerstein, M., & Zhao, H. (2004). Information assessment on predicting protein-protein interactions. *BMC Bioinformatics*, 5. <https://doi.org/10.1186/1471-2105-5-154>
- Lipton, Z. C., Berkowitz, J., & Elkan, C. (2015). *A Critical Review of Recurrent Neural Networks for Sequence Learning*. <http://arxiv.org/abs/1506.00019>

- Mahapatra, S., & Sahu, S. S. (2021). Improved prediction of protein-protein interaction using a hybrid of functional-link Siamese neural network and gradient boosting machines. *Briefings in Bioinformatics*, 22(6). <https://doi.org/10.1093/bib/bbab255>
- Mrozek, D., Socha, B., Kozielski, S., & Małysiak-Mrozek, B. (2016). An efficient and flexible scanning of databases of protein secondary structures: with the segment index and multithreaded alignment. *Journal of Intelligent Information Systems*, 46(1), 213–233. <https://doi.org/10.1007/s10844-014-0353-0>
- Nanni, L., Lumini, A., & Brahnam, S. (2014). An empirical study of different approaches for protein classification. *Scientific World Journal*, 2014. <https://doi.org/10.1155/2014/236717>
- Oughtred, R., Stark, C., Breitkreutz, B. J., Rust, J., Boucher, L., Chang, C., Kolas, N., O'Donnell, L., Leung, G., McAdam, R., Zhang, F., Dolma, S., Willems, A., Coulombe-Huntington, J., Chatr-Aryamontri, A., Dolinski, K., & Tyers, M. (2019). The BioGRID interaction database: 2019 update. *Nucleic Acids Research*, 47(D1), D529–D541. <https://doi.org/10.1093/nar/gky1079>
- Pan, A., Lahiri, C., Rajendiran, A., & Shanmugham, B. (2016). Computational analysis of protein interaction networks for infectious diseases. *Briefings in Bioinformatics*, 17(3), 517–526. <https://doi.org/10.1093/bib/bbv059>
- Peter J. Russell. (2009). *iGenetics: A Molecular Approach* (3rd ed.). Pearson.
- Pnas, K. (2007). *Predicting protein-protein interactions based only on sequences information* (Vol. 104, Issue 11).
- Puig, O., Caspary, F., Rigaut, G., Rutz, B., Bouveret, E., Bragado-Nilsson, E., Wilm, M., & Séraphin, B. (2001). The tandem affinity purification (TAP) method: A general procedure of protein complex purification. *Methods*, 24(3), 218–229. <https://doi.org/10.1006/meth.2001.1183>

- Rao, V. S., Srinivas, K., Sujini, G. N., & Kumar, G. N. S. (2014). Protein-Protein Interaction Detection: Methods and Analysis. *International Journal of Proteomics*, 2014, 1–12. <https://doi.org/10.1155/2014/147648>
- Richoux, F., Servantie, C., Borès, C., & Téletchéa, S. (2019). *Comparing two deep learning sequence-based models for protein-protein interaction prediction*. <http://arxiv.org/abs/1901.06268>
- Salinas, V. H., & Ranganathan, R. (2018). Coevolution-based inference of amino acid interactions underlying protein function. *ELife*, 7, e34300. <https://doi.org/10.7554/eLife.34300>
- Smallridge, R. (2003). How to see the big picture. *Nature Reviews Molecular Cell Biology* 2003 4:12, 4(12), 909–909. <https://doi.org/10.1038/nrm1274>
- Sun, T., Zhou, B., Lai, L., & Pei, J. (2017). Sequence-based prediction of protein protein interaction using a deep-learning algorithm. *BMC Bioinformatics*, 18(1). <https://doi.org/10.1186/s12859-017-1700-2>
- TANFORD Vol, C., & Charles Tanford, B. (1961). Contribution of Hydrophobic Interactions to the Stability of the Globular Conformation of Proteins. In 19.iZ). *Acad. Sci* (Vol. 49, Issue 2).
- Thomas, P. D., Campbell, M. J., Kejariwal, A., Mi, H., Karlak, B., Daverman, R., Diemer, K., Muruganujan, A., & Narechania, A. (2003). PANTHER: A library of protein families and subfamilies indexed by function. *Genome Research*, 13(9), 2129–2141. <https://doi.org/10.1101/gr.772403>
- Voulodimos, A., Doulamis, N., Doulamis, A., & Protopapadakis, E. (2018). Deep Learning for Computer Vision: A Brief Review. In *Computational Intelligence and Neuroscience* (Vol. 2018). Hindawi Limited. <https://doi.org/10.1155/2018/7068349>
- Xenarios, I., Salwinski, D., Duan, J., Higney, P., Kim, S.-M., & Eisenberg, D. (2002). DIP, the Database of Interacting Proteins: a research tool for studying cellular networks of protein interactions. In *Nucleic Acids Research* (Vol. 30, Issue 1).

- Yang, X., Zhang, Z., & Wuchty, S. (2021). Multi-scale convolutional neural networks for the prediction of human-virus protein interactions. *ICAART 2021 - Proceedings of the 13th International Conference on Agents and Artificial Intelligence*, 2, 41–48. <https://doi.org/10.5220/0010185300410048>
- You, Z. H., Lei, Y. K., Zhu, L., Xia, J., & Wang, B. (2013). Prediction of protein-protein interactions from amino acid sequences with ensemble extreme learning machines and principal component analysis. *BMC Bioinformatics*, 14(SUPPL8). <https://doi.org/10.1186/1471-2105-14-S8-S10>
- You, Z. H., Zhu, L., Zheng, C. H., Yu, H. J., Deng, S. P., & Ji, Z. (2014). Prediction of protein-protein interactions from amino acid sequences using a novel multi-scale continuous and discontinuous feature set. *BMC Bioinformatics*, 15. <https://doi.org/10.1186/1471-2105-15-S15-S9>
- Zahiri, J., Yaghoubi, O., Mohammad-Noori, M., Ebrahimpour, R., & Masoudi-Nejad, A. (2013). PPIevo: Protein-protein interaction prediction from PSSM based evolutionary information. *Genomics*, 102(4), 237–242. <https://doi.org/10.1016/j.ygeno.2013.05.006>
- Zhang, L., Yu, G., Xia, D., & Wang, J. (2019). Protein–protein interactions prediction based on ensemble deep neural networks. *Neurocomputing*, 324, 10–19. <https://doi.org/10.1016/j.neucom.2018.02.097>
- Zhou, Y. Z., Gao, Y., & Zheng, Y. Y. (2011). Prediction of Protein-Protein Interactions Using Local Description of Amino Acid Sequence. In *CCIS* (Vol. 202).
- Zhu, H. J., You, Z. H., Shi, W. L., Xu, S. K., Jiang, T. H., & Zhuang, L. H. (2019). Improved Prediction of Protein-Protein Interactions Using Descriptors Derived From PSSM via Gray Level Co-Occurrence Matrix. *IEEE Access*, 7, 49456–49465. <https://doi.org/10.1109/ACCESS.2019.2907132>