國立臺灣海洋大學

資訊工程學系

碩士學位論文

指導教授：阮議聰 博士

預測超級細菌綠膿桿菌蛋白質間的交互作用

Predicting Protein-Protein Interactions of Superbug Pseudomonas Aeruginosa

研究生：張祐琪 撰

中華民國 113 年 6 月

預測超級細菌綠膿桿菌蛋白質間的交互作用

Predicting Protein-Protein Interactions of Superbug Pseudomonas Aeruginosa

研 究 生：張祐琪　　Student：Yo Chi Chang
指導教授：阮議聰　　Advisor：Eric Y.T.Juan

國 立 臺 灣 海 洋 大 學
資 訊 工 程 學 系
碩 士 論 文

A Thesis
Submitted to Department of Computer Science and Engineering
College of Electrical Engineering and Computer Science
National Taiwan Ocean University
In Partial Fulfillment of the Requirements
for the Degree of
Master of Science
in
Computer Science and Engineering
June 2024
Keelung, Taiwan, Republic of China

中華民國 113 年 6 月

摘要

超級細菌擁有多重抗生素的抗藥性，有的甚至對所有抗生素都具有抗藥性，導致感染的病患可能會面臨沒有藥物治療的狀況，這樣的多重抗藥性細菌慢慢地受到更多醫學、科學相關人士及社會大眾的重視，也因此本論文使用了被列為超級細菌的綠膿桿菌作為測試資料。在生物界中蛋白質之間交互作用是必不可少的，所以我們主要是透過綠膿桿菌的蛋白質交互作用來做這次實驗。現今運用計算機預測分析蛋白質的交互作用已是生物資訊學的熱話題，相比以往透過大量實驗檢測來分析，不管是時間或是金錢成本皆是一大問題，而計算機方法和機器學習能夠快速分析蛋白質之間的交互作用，使用不同方法加速計算機的分析時間以及提升預測準確度。

本篇論文中，我們使用了蛋白質交互作用預測系統，再結合不同的蛋白質描述方法，有協方差(AC)和簡化胺基酸(RAAA)兩種，將蛋白質交互作用對的數值陣列轉換成特徵向量，最後使用三種機器學習的方法包括 KNN、SVM 及 DNN，在 DNN 中又會再分為加入 DropOut 層，以及無 DropOut 層的 DNN_ND 來做訓練並進行預測。最終實驗結果，最高準確率是在使用 Ushuffle 偽造非交互作用蛋白質對方法，蛋白質描述方法 RAAA2-lt3 及機器學習模型 DNN_ND 的 95.6%。最高精確率是在使用 Ushuffle 偽造非交互作用蛋白質對方法，蛋白質描述方法 RAAA1-lt2 及機器學習模型 DNN 的 96.3%。最高靈敏度是在使用 Ushuffle 偽造非交互作用蛋白質對方法，蛋白質描述方法 RAAA1-lt2、RAAA2-lt3 及機器學習模型 DNN_ND 的 95.6%。

關鍵字：蛋白質交互作用，蛋白質描述方法，深度神經網絡

Abstract

Superbugs possess resistance to multiple antibiotics, with some even exhibiting resistance to all available antibiotics. This situation can leave infected patients without effective treatment options. The increasing recognition of these multi-drug resistant bacteria by medical and scientific communities, as well as the general public, underscores the importance of this issue. Consequently, this study employs Pseudomonas aeruginosa, a known superbug, as the test subject. Protein-protein interactions (PPIs) are crucial in biological systems, and thus, this study primarily focuses on PPIs in Pseudomonas aeruginosa. The computational prediction and analysis of PPIs have become a hot topic in bioinformatics. Compared to the traditional extensive experimental testing, computational methods and machine learning can rapidly analyze PPIs, significantly reducing both time and financial costs while enhancing prediction accuracy.

In this thesis, we utilized a PPI prediction system combined with different protein descriptor methods, specifically Auto Covariance (AC) and Reduced Amino Acid Alphabet (RAAA). These methods convert the numerical arrays of PPI pairs into feature vectors. Subsequently, three machine learning approaches—K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Deep Neural Networks (DNN)—were employed for training and prediction. The DNN model was further divided into two variants: one with DropOut layers (DNN) and one without (DNN_ND).The experimental results demonstrated that the highest accuracy rate of 95.6% was achieved using the Ushuffle method for generating non-interacting protein pairs, the RAAA2-lt3 protein descriptor method, and the DNN_ND model. The highest precision rate of 96.3% was obtained using the Ushuffle method, the RAAA1-lt2 protein descriptor method, and the DNN model. The highest sensitivity of 95.6% was achieved using the Ushuffle method, the RAAA1-lt2, and RAAA2-lt3 protein descriptor methods, and the DNN_ND model.

Keywords: Protein-protein interaction, protein description method, deep neural network

目錄

圖目錄

表目錄

第一章 緒論

1.1 研究背景與動機

超級細菌（superbug）的正式名稱為「抗藥性細菌」，指的是細菌對多種抗生素具有抗藥性的情況。雖然超級細菌並非無懈可擊，但要克服它卻相當困難。抗生素對細胞壁、細胞膜、蛋白質或核酸合成的抑制機制理論上應該會殺死絕大多數細菌。然而，自然界的生物在面對環境變化時通常會尋找新的生存之道，細菌亦不例外。有些細菌能夠透過基因突變或質體交換等方式改變自身基因，以發展出對抗抗生素的能力[1]。這些被稱為「超級細菌」的細菌擁有多重抗藥性（multidrug resistance）[2]，通常對三種或三種以上的抗生素具有抗藥性。

蛋白質(protein)是生物體細胞的重要成分，扮演著多種關鍵角色。它們是由一個或多個 α-胺基酸殘基長鏈條組成的一個大型生物分(Large Biomolecule)或高分子（Macromolecule）[3]。蛋白質的功能由其理化性質以及結構而定，這些功能決定了蛋白質之間如何互相作用[4]，並在生物體內完成維持生命的化學反應。不同的氨基酸通過脫水縮合(dehydration condendensation)[5]形成若干個肽鍵從而組成一個肽，多個肽進行多級摺疊後組成一個蛋白質分子，故蛋白質也稱為多肽[6]。每個蛋白質都有獨特的 α-胺基酸序列，而這種序列的排列方式是由相應的基因編碼所決定的。除了遺傳密碼所編碼的 20 種標準胺基酸(Amino acids) [7]外，有些 α-胺基酸殘基還可以通過原子排序的改變而產生不同的化學結構，進而影響蛋白質的功能和調節。蛋白質的多樣性源於其胺基酸序列和立體結構的組合方式[8]。這些結構以一級、二級、三級和四級的形式存在。一級結構(primary structure)：指蛋白質中胺基酸殘基的排列順序，這決定了每種蛋白質的獨特性。二級結構(secondary structure)：則是通過氫鍵的形成來維持，包括 α-螺旋和 β-折疊等形式。三級結構(tertiary structure)：是在二級結構的基礎上進一步形成，主要受到氨基酸側鏈之間的疏水交互作用，氫鍵，凡得瓦力和離子鍵維持的。四級結構(quaternary structure)：是多個具有三級結構的多肽以特定方式組合形成的三維結構[9]。

蛋白質在人體生理活動中也非常重要，包括生長、發育、遺傳、繁殖和運動等。這些生命活動無不依賴於蛋白質的參與，因此，人體內許多物質與蛋白質密切相關。舉例而言，胺類、神經傳遞物質、多肽類激素、抗體、腶、核蛋白以及在血液中起載體作用的蛋白質，均對於調節生理功能和維持新陳代謝起到關鍵作用。人體必須獲取食物中的蛋白質並經由腸胃道消化，將其分解為胺基酸，才能被身體吸收利用。根據營養學的分類，胺基酸可分為兩類，即必須胺基酸和非必須胺基酸。必須胺基酸[10]是指人體無法自身合成或合成量不足以滿足需求，因

此必須從食物中攝取的胺基酸，共計九種。例如，苯丙胺酸(Phenylalanine)可用於傳達大腦和神經細胞之間的化學訊息，同時具有天然的止痛作用。纈胺酸(Valine)則可促進肌肉新陳代謝和組織修復，並維持氮的平衡。蘇胺酸(Threonine)是膠原蛋白和牙齒細胞細胞質中的重要組成部分，且具有防止肝臟脂肪積聚的作用。色胺酸(Tryptophan)有助於促進睡眠，減輕疼痛和頭痛，並減輕焦慮和緊張情緒。異白胺酸(Isoleucine)則是形成血紅蛋白的必需胺基酸，具有穩定和調節血糖和能量的功效。白胺酸(Leucine)可促進骨骼、皮膚和肌肉組織的修復，並穩定血糖濃度。甲硫胺酸(Methionine)有助於促進消化系統功能，解除有害物質的毒性，並幫助衰竭的肌肉恢復。離胺酸(Lysine)對於人體蛋白質的構建至關重要，可促進發育、組織修復和抗體、荷爾蒙和酵素的產生，並增強注意力。組胺酸(Histidine)有助於製造紅血球和白血球，提高身體的免疫反應。

另一方面，非必須胺基酸[11]是人體可以自身合成或由其他胺基酸轉化而得到的胺基酸，共計十一種。例如，精胺酸(Arginine)能夠維持腦下垂體的正常功能，並促進體內尿素的生成，以降低血氨濃度。甘胺酸(Glycine)在免疫系統合成中扮演著重要角色，協助從血液中釋放氧氣至組織細胞，同時促進荷爾蒙的合成，增強免疫功能。麩醯胺酸(Glutamine)則有助於提高腦部功能，維持腦部正常的生理運作。酪胺酸(Tyrosine)是腦神經傳遞的重要物質之一，能夠協助克服憂鬱，改善記憶。苯丙胺酸(Alanine)在體內蛋白質合成過程中扮演關鍵角色，有助於產生抗體，並協助葡萄糖和有機酸的代謝。絲胺酸(Serine)有助於肌肉和肝臟存儲肝糖，同時協助抗體的合成，以及神經纖維外鞘的合成。胱胺酸(Cystine)有助於皮膚再生，對於治療燒傷和手術後傷口的癒合至關重要。半胱胺酸(Cysteine)能夠幫助清除細胞中的有毒物質，保護細胞免受損害。脯胺酸(Proline)則能夠加快傷口的癒合，是氨基酸輸液中的重要成分。天門冬胺酸(Aspartic acid)能夠維持中樞神經系統的平衡，使人免受過度緊張或過度鎮定的影響。瓜胺酸(Citrulline)能夠促進能量的產生，刺激免疫系統，並清除可能對活細胞造成損害的氨毒。此外，蛋白質也構成了人體組織器官和支架的主要物質。蛋白質攝取不足可能導致肌肉消瘦、免疫力下降和貧血等問題，而攝取過量則可能引起骨質疏鬆和腎臟負擔過重等不良影響。

近年來已有不少學者在研究有關計算機預測蛋白質交互作用的課題。早期透過實驗發現蛋白質交互作用是一件耗時複雜的工作，所以當時的蛋白質研究都侷限在單一蛋白質的特性，或是以幾個少數蛋白質之間的交互作用來研究。一般使用來檢測蛋白質間交互作用的方法，例如：免疫沉澱法(Immunoprecipitation)、染色質免疫沉澱(Chromatin Immunoprecipitation)、螢光共振能量傳輸(fluorescence resonance energy transfer, FRET)、表面電漿共振(surface plasma resonance, SPR)等技術，儘管在預測上各有其優點，不過還是存在著如需要可連結蛋白質的特定抗體、需要大量的樣品進行實驗、實驗過程繁瑣、需要耗費大量的時間來完成等缺

點。近二十幾年來，隨著生物科技技術的提升，已有許多蛋白質根據實驗結果加入特性描述，並建立了許多蛋白質資料庫，也有許多檢測蛋白質交互作用實驗的結果被加入且建立了蛋白質交互作用資料庫，利用蛋白質及蛋白質交互作用資料庫所提供之大量資料，使得研發較不花費成本又快速可靠的檢測蛋白質交互作用方法的研究變得更加重要。

用機器學習的方法對蛋白質的交互作用進行預測[12]，首先要解決的問題即蛋白質對的編碼問題，即將蛋白質對轉化成對應的特徵向量，如果能使用較好的蛋白質編碼，可以提升蛋白質交互作用的預測表現，近年來已經有眾多學者提出不同的蛋白質描述方法，這些方法大致上分成兩類：(1)以序列組成為基礎，此類方法以蛋白質之胺基酸序列組成(Amino Acid Composition, AAC) [13]，此類型描述方法包含胺基酸組成的百分比，並加入其他蛋白質的特性作為額外註記。(2)以相似性為基礎，此類別方法即是以蛋白質的胺基酸順序性為基礎。本論文所使用的為第一種蛋白質描述方法，分別為群集胺基酸(Reduced Amino Acid Alphabets, RAAA)以及協方差(Auto covariance, AC)[14]。並利用 tensorflow 建立一個深度神經網路(Deep Neural Network)和 KNN、SVM 分別對蛋白質的交互作用進行預測，並比較不同的描述方法以及使用不同的機器學習的方法，如何才能提升預測蛋白質交互作用的準確性。

1.2 論文架構

在本論文中，第二章介紹幾個蛋白質交互作用的相關研究。第三章介紹本論文所使用的蛋白質描述方法及偽造非交互作用蛋白質對方法。第三章第一部份簡單說明此篇所採用的實驗資料集來源，第二部分介紹本論文使用的兩種偽造非交互作用蛋白質對方法，第三部份會介紹幾種在預測系統中會使用的蛋白質描述方法。在第四章特別把機器學習分類器拉出來介紹，最後會詳細說明實驗所使用到的資料和使用方法。第五章會依實驗結果數據比較偽造非交互作用蛋白質對及蛋白質描述方法搭配不同機器學習預測方法的優劣。最後在第六章陳述本篇論文的結論。

第二章 相關研究

2.1 基於實驗的蛋白質互相作用的研究

2.1.1 酵母菌雙雜合系統(yeast two-hybrid system)

最早是由 Fields and Song 使用出這項技術來研究蛋白質間的交互作用，他們一開始使用的是釀酒酵母(Saccharomyces cerevisiae)的 GAL4[15]表達系統的特性來對蛋白質和 DNA 進行交互作用，這秀研究在大規模的蛋白質交互作用上酵母菌雙雜合技術[16]扮演了更加重要的角色，不過此種方法雖然靈敏度高，偽陽性出現率也高。

2.1.2 純化技術(tandem affinity purification, TAP)

純化技術是由 Rigaut[17]等人開發出來用來純化蛋白質複合體的技術，利用特殊設計的蛋白質標籤基因將兩個不同的親和標籤連接到目標蛋白質上，經多層次的液相層析法後得到接近自然狀態的蛋白質複合物。此種方法假陽性率跟假陰性率低，可以分析穩定複合物中蛋白質交互作用，能有效避免酵母菌雙雜合系統、免疫沉澱法假陽性率過高的問題。

2.1.3 GST pull-down 技術

GST pull-down 技術利用 GST 標籤蛋白和目標蛋白充當一種誘餌蛋白融合後，可以由融合蛋白中得到與目標蛋白有交互作用的蛋白。其優點為操作簡便，可直接檢測是否有交互作用。缺點是只能檢測穩定或者強烈的交互作用，以及蛋白互相作用會受到內源性蛋白質的干擾。Guttman 等[18]利用該技術發現了 $NPXY_{4507}$序列與患有阿茲海默症的小鼠腦部蛋白發生交互作用。

2.1.4 免疫共沉澱法(co-immunoprecipitation)

免疫共沉澱法[19]是蛋白質交互作用研究上運用廣泛且可信度高的方法，原理將蛋白質視為抗原，並利用抗體會與抗原特異性結合的特性來進行分析。這個方法可以將含有上千種不同蛋白質的樣品，然後以抗體和抗原之間的專一性作用為基礎，分離和濃縮出特定蛋白質。該方法研究的蛋白質交互作用都是在自然的

狀態下進行，細胞內裡完成，可以避免人為和外界因素的影響。不過此方法靈敏度不高，可能檢測不到低親和力和瞬間的蛋白質交互作用。

2.1.5 蛋白質晶片(protein chip)

此種方法以蛋白質為生物探針，排列在晶片上，藉由抗原-抗體的免疫反應用以進行蛋白質交互作用的檢測，是一種高通量的蛋白功能分析技術。Zhu 等[20]以及 Ptacek[21]皆利用了蛋白質晶片的技術發現了眾多會產生交互作用的蛋白。此種方法具有操作簡單且容易標準化操作，對於單一或是整體功能性單白質體研究都具有相當的助益。

2.2 基於計算方法的蛋白質交互作用的研究

基於計算方法預測蛋白質的交互作用一部分是依據蛋白質的基因組訊息(genomic information)，像是 protein phylogenetic profiles [22]，基因鄰接[23]，基因融合[24]-[26]，使用蛋白質結構訊息的方法[27][28]，交互作用的蛋白質之間的序列保守性[29][30]，但是這些方法都不能用來判斷蛋白質的可靠度。

2.3 基於胺基酸的理化性質的研究

Nanni 等[31]提出了一種 HKNN(K-local hyperplane distance nearest neighbor)分類法，每個 HKNN 可以藉由不同胺基酸的理化性質來訓練。Guo 等[32]提出用協方差(auto covariance, AC)對蛋白質進行編碼，並利用 SVM(Support Vector Machine)對蛋白質的交互作用進行預測，在胺基酸的理化性質分別用了疏水性(hydrophobicity)、親水性(hydrophicility)、側鏈的質量(volumes of side chains of amino acids)、極性(polarity)、極化性(polarizability) 、暴露表面積(solvent-accessible surface area)以及側鏈的淨電荷指數(net charge index of side chains of amino acids)，他們對釀酒酵母的蛋白質交互作用資料進行 5 次交叉驗證測試法(5 fold cross validation)，其最後結果表現很好。

第三章 蛋白質資料收集與處理

3.1 測試資料集合

本論文實驗所使用的資料集為 STRING(Search Tool for the Retrieval of Interacting Genes/Proteins)資料庫內綠膿桿菌的蛋白質序列對，以及綠膿桿菌的蛋白質互相作用資料。目前常使用一些知名蛋白質交互作用資料庫[33]-[36]的資料套用機器學習的方法於蛋白質交互作用的預測研究中。

STRING 資料庫蒐集了上萬個物種，包含了超過五千多萬筆的蛋白質資料，不僅有蛋白質序列對，蛋白質交互作用的資料在網站上也都有紀錄。蛋白質交互作用是細胞在進行生化反應的一個重要依據，蛋白質交互作用可以用來評估功能性基因的相關資料並且提供一個可以用來標註蛋白質結構、功能以及進化特性的平台。STRING 是利用兩兩蛋白質之間的相關性把整個蛋白質集合做網路性的描述，這些關聯度的資料來源包含有四種：(1)基因脈絡(Genomic Context)、(2)高通量實驗(High-throughput Experiments)、(3)蛋白質共表現(Protein Coexpression)、還有(4)其它已經被發表出來的研究文獻或是其它資料庫所蒐集的交互作用資料，裡面包含了幾個常見的資料庫，例如：MINT、HPRD、BIND、DIP、BioGRID、IntAct…等。STRING 也會對每筆關聯度資料依照可靠度做評比整合，提供一組可信度分數，分數從 0 到 1000，1000 代表可信度最高，藉以提供學者更多資訊取用合適的資料。

本篇論文將以 STRING(V11.5)較新版本的綠膿桿菌當做機器學習的測試集(test set)，和 STRING(V11.0)版本的資料集當做訓練集(train set)，其中訓練集(train set)的資料從 STRING(V11.5)和 STRING(V11.0)取交集的交互作用對，來評估不同的胺基酸理化性質在蛋白質對描述方法優劣。

3.1.1 蛋白質不交互作用資料集合

不交互作用蛋白質對(Negative example)作為訓練集(Train set)的一部份，對是否能正確且成功的使預測系統有效率的預測致關重要，本論文產生偽造不交互作用蛋白質對方法有兩種，第一種本論文實驗名稱為 Ushuffle[37]，是以不同的亂序演算法重新編排原始蛋白質的胺基酸序列(Amino acids sequence)偽造成新的蛋白質序列，並給予偽造的蛋白質名稱，和原始蛋白質配對成偽造不交互作用。第二種實驗名稱為 LeftRight [38]，是將蛋白質交互作用對(Protein Interacting Pairs)的集合中重新隨機配對成偽造不交互作用的蛋白質對，並保證每對都是唯一。

於上述兩種蛋白質對，為了去除重複且具同源相似性(homology bias)的蛋白質資料，皆透過以下的處理：
(1)交互作用訓練和測試集合再互取差集，使訓練集和測試集相互獨立。
(2)因為少於 50 個胺基酸的序列有可能只是蛋白質中的片段(fragment)，為了避免這種情況發生，將包含這種片段的蛋白質對都去除。
(3)刪除重複的蛋白質相作用對，假設資料集有一對蛋白質交互作用對 Φ1(A,B)包含蛋白質 A 和蛋白質 B，並有另一對蛋白質交互作用對 Φ2(B,A)包含蛋白質 B 和蛋白質 A，Φ1 和 Φ2 將被視為重複交互作用對，刪除其中一對只保留一對。
(4)去除相似的蛋白質，本論文分別使用 CD-Hit[39]將訓練集(Train Set)和測試集(Test Set)的蛋白質交互作用對做相似度的評估刪除冗餘的蛋白質，將相似度設定為 40，相似度的評估方式將依照聚類定義將蛋白質進行聚類。

本篇論文將以這個資料集為基礎，來評估蛋白質及蛋白質對描述方法優劣，並建立預測分類模組。之後再將 STRING 資料庫中最新資料作為此模組的輸入測試，驗證我們提出的新系統的可信度。

3.1.2 Ushuffle 資料集合

Ushuffle[37]是由 Minghu Jiang 所提出的一種偽造非交互作用蛋白質對的方法，這個方法是 Euler 算法的延伸，用不同的亂序演算法去將原本的蛋白質胺基酸序列順序打亂，偽造成新的蛋白質序列並且給予名稱，偽造出來的蛋白質和原始的蛋白質是無法有相互作用的，可以透過更改其中的 k-let 大小偽造出不同的蛋白質，在本實驗中，會使用 K1 的參數值來偽造非交互作用蛋白質對。

3.1.3 LeftRight 資料集合

LeftRight[38]方法根據交互作用蛋白質對中兩個蛋白質序列之間的最短路徑越長，則交互作用的機率越低的假設，重新將它們隨機配對，並確保蛋白質對都是唯一的，以產生偽造的不交互作用蛋白質對。

3.2 常見的蛋白質描述方法

在蛋白質分類與預測問題中，由於每一條的蛋白質長度不一，所以適當的描述蛋白質非常重要。為了分類器的預測，必須把每一條蛋白質序列轉換成固定長度的特徵向量，並保存序列中的蛋白質資訊。蛋白質描述方法有非常多種，一個好的描述方法能大為增加預測的準確性，並且可以幫助分析蛋白質組成與特性。

以下介紹常見的幾種蛋白質序列特徵描述方法。

3.2.1 偽胺基酸組成 (Pseudo-Amino Acid Composition, PseAA)

是由 Chou[40]所提出的蛋白質描述方法，此方法又稱為 Type 1 PseAA，該方法主要是以蛋白質的胺基酸組成(Amino Acid Composition ,AAC)為基礎加上部份的胺基酸的理化性質。蛋白質的胺基酸組成(AAC)在蛋白質的描述方法裡極具代表性，此方法是用來計算單一的蛋白質序列中 20 種標準胺基酸的組成百分比，這種蛋白質描述方法計算簡單且容易分析，但準確率通常不高。此方法一般能夠描述出整個蛋白質的特性，不容易描述出蛋白質各個區域分別的情況。

3.2.2 雙肽組成 (Dipeptide Composition, Dipeptide)

肽，又稱縮氨酸，介於胺基酸和蛋白質之間的物質。雙肽組成是計算單一條蛋白質序列中，兩個 20 種標準胺基酸配對組合佔全部配對組合的百分比，例如 AA、AC、AD…EE 等雙肽組合在一條蛋白質序列中出現的次數，總共會有 400 種雙肽，所以會產生 400 維的特徵向量 F，如(式 1)。給定一個蛋白質 P，序列長度 L，序列$R_1R_2R_3R_4R_5\ldots R_L$，將數值 1,2,…,20 分別表示成蛋白質序列中的 20 種標準胺基酸 (A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, and Y)。R_nR_{n+1}為一雙肽組合，n=1,2,…,L-1，R_n胺基酸轉成數值 i，R_{n+1}胺基酸轉成數值 j，透過(式 2)計算全部雙肽組合得到特徵向量。

$$F = \begin{bmatrix} F_1 \\ F_2 \\ \cdots \\ F_{(i-1)*20+j} \\ \cdots \\ F_{400} \end{bmatrix} \quad (式\ 1)$$

$$F_{(i-1)*20+j} = \frac{total\ number\ of\ dipeptide\ i,j}{L-1} \quad (式\ 2)$$

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

3.2.3 協方差(Auto covariance, AC)

在描述蛋白質序列中，我們時常用協方差(Auto covariance, AC)來說明一條蛋白質序列殘基和他鄰居之間的關係[32]。在本篇論文中協方差主要是用來把數值陣列轉化成統一的特徵相量，(式 3)

$$AC_{lag,j} = \frac{1}{L-lag}\sum_{i=1}^{L-lag}(P_{i,j} - \frac{1}{L}\sum_{i=1}^{L}P_{i,j}) \times (P_{(i+lag),j} - \frac{1}{L}\sum_{i=1}^{L}P_{i,j}) \quad (式\ 3)$$

其中 i 代表在蛋白質序列 P 中的位置，j 代表一個描述符(descriptor)，L 表示蛋白質序列 P 的長度。AC 的數量我們可以用 D 來表示，可以根據(式 4)。

$$D = lg \times p \quad (式\ 4)$$

其中lg表示lag的最大值($lag = 1,2,\dots,lg$)，p代表描述符的總數。

3.2.4 簡化氨基酸 (Reduced Amino Acid Alphabets, RAAA)

蛋白質序列的資料相比蛋白質結構資料量大得多。許多非相關序列可以採用類似的 3D 折疊，因此能夠在類似的 3D 結構中找到不同種類的胺基酸，通過將 20 個氨基酸組合成具有相似特徵數量也較少的代表性殘基，可以簡化蛋白質序列。

20 種標準氨基酸中有些氨基酸的物理化學特性是相似或相同的，因此 20 種標準氨基酸可以再被分類成更少種的氨基酸，此種方法即為簡化胺基酸(Reduced Amino Acid Alphabets, RAAA) [41]。比起傳統的 20 種標準胺基酸，使用簡化胺基酸來描述蛋白質序列，不僅簡化了系統的複雜度，也保留了蛋白質區域結構訊息的能力[42]。本論文所使用的為 PT20、CP13、CP11，前三種的簡化胺基酸，在後面實驗分別以 RAAA1、RAAA2、RAAA3 代稱。

本論文使用簡化胺基酸選取四種不同長度肽的蛋白質描述法，分別為:RAAA1_lt2 代表使用第一種簡化胺基酸，並選取長度 2 的肽(Peptide)合併起來，形成維度總共為 400 的特徵向量。RAAA2_lt2 代表使用第二種簡化胺基酸，並選取長度 2 的肽(Peptide)合併起來，形成維度總共為 169 的特徵向量。RAAA2_lt3 代表使用第三種簡化胺基酸，並選取長度 3 的肽(Peptide)合併起來，形成維度總共為 2197 的特徵向量。RAAA3_lt3 代表使用第三種簡化胺基酸，並選取長度 3 的肽(Peptide)合併起來，形成維度總共為 1331 的特徵向量。

以上僅列出一條蛋白質序列透過簡化胺基酸描述方法形成的維度，在本篇論文實驗中須使用兩條蛋白質預測交互作用，則每個維度都需再乘以 2 倍代表兩條，舉例說明 RAAA1_lt2：一條蛋白質序列下維度為 400，兩條蛋白質序列則 800；RAAA2_lt2：一條蛋白質序列下維度為 169，兩條蛋白質序列則為 338，以下簡化胺基酸以此類推。

(表 1)：簡化氨基酸 RAAA 在不同長度肽(Peptide)底下的維度

Cluster profiles	Feature vector dimension of n-peptide composition with different cluster profiles			
	n=1	n=2	n=3	n=4
PT20	20	400	-	-
CP13	13	169	2197	-
CP11	11	121	1331	-

(表 2)：蛋白質對在 RAAA 的特徵向量維度

Method	Feature vector dimension
RAAA1_lt2	800
RAAA2_lt2	338
RAAA2_lt3	4394
RAAA3_lt3	2662

第四章 研究方法

4.1 機器學習(Machaine Learning)

機器學習被視為人工智慧(AI)的一項技術，與傳統的程序不同。它通過處理大量的數據並進行學習，以歸納推理的方式解決問題。當新的數據出現時，機器學習模型能夠自動更新對問題的理解，從而調整對原有問題的認識。

機器學習的定義可以解釋為:「透過過去分析的資料和經驗中學習，從當中找出其運行規則，最終實現人工智慧的方法。」該技術已廣泛應用於多個領域，如資料探勘、電腦視覺、自然語言處理、DNA 序列測序、語音和手寫辨識等。

機器學習的流程可以分為以下七個步驟:
(1)收集數據（Gathering data）
(2)準備數據（Preparing that data）
(3)選擇模型（Choosing a model）
(4)訓練模型（Training）
(5)評估模型（Evaluation）
(6)調整參數（Hyperparameter tuning）
(7)預測推論（Prediction）

4.1.1 Tensorflow

Tensorflow 是由 Google Brain 的團隊開發，最初是用在 Google 內部的研究和生產，是一個機器學習的框架，可在大規模和異構環境中運行，使用數據流圖來表示計算 Tensor-Flow，共享狀態以及改變該狀態的操作，將數據流圖的節點映射到集群中的多台計算機上，並映射到跨多個計算設備，包括多核 CPU，通用 GPU 和定制設計的 ASIC（稱為張量處理單元 TPU）[43]。TensorFlow 支持多種應用，著重在深度神經網絡的訓練和推理。目前已廣泛用於機器學習研究，例如: 語音辨識、圖片識別、翻譯. . 等。TensorFlow 的計算過程會用有狀態的資料流圖來表示，名字源自於這類神經網路對多維陣列執行的操作，而這些多維陣列被稱為張量(Tensor)。

4.1.2 深度學習(人工神經網路)

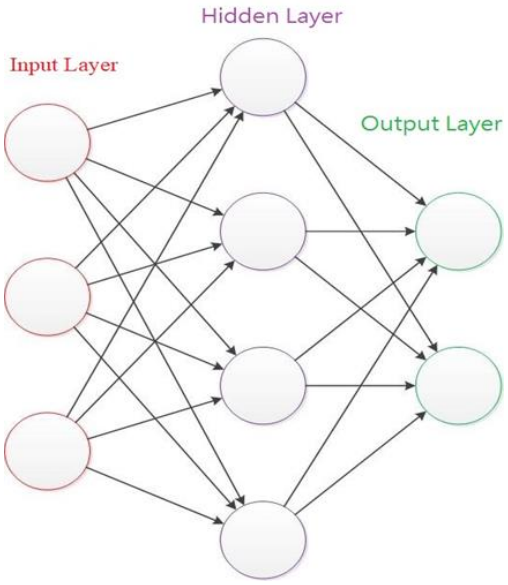
深度學習(Deep Learning，以下簡稱 DL)是機器學習的的延伸，由多層的人工神經網路作為架構，會像人類大腦運作方式從大量資料中進行特徵學習的演算法，讓系統直接對非結構化和未做標記的資料學習和做決策。

在深度學習領域中，其優勢主要體現在高效的演算法上，這些演算法包括無監督學習、半監督學習以及分層擷取特徵，這些演算法的應用可以取代傳統的人工手動標記特徵資料的方式。特徵學習的主要目的在於尋求更優的表示方法和模型，透過大量未標記資料的學習，以達到更精準的模型表現。這些表示方法源自於神經科學的啟發，並且建立在類似神經系統中的資訊處理和通訊模式的理解上，例如神經編碼，旨在捕捉神經元反應之間的關係，以及大腦中神經元電活動之間的交互作用。

此外，學界中有多種深度學習框架，如深度神經網絡（Deep Neural Networks, DNNs）是一種基於多層神經元的結構，其隱含層的深度對於模型的性能至關重要。卷積神經網絡（Convolutional Neural Networks, CNNs）在處理圖像和視覺任務中表現出色，它們能夠有效地捕捉圖像中的空間結構信息。另外，遞歸神經網絡（Recurrent Neural Networks, RNNs）特別適用於處理序列數據，例如語言模型和時間序列預測。

這些深度學習方法的應用非常廣泛，涵蓋了圖像處理、語音識別、自然語言處理、生物信息學等各個領域，並在許多任務中取得了顯著的成果。深度學習也只要三個步驟：**建構網路、設定目標、開始學習。**

一個經典的神經網路包含三個層次，1. 輸入層，2. 隱藏層，3. 輸出層。如 (圖 1)。將訓練資料透過建立的訓練模型不斷優化，最後完成訓練模型，再將我們的測試資料輸入比對，測出準確性。

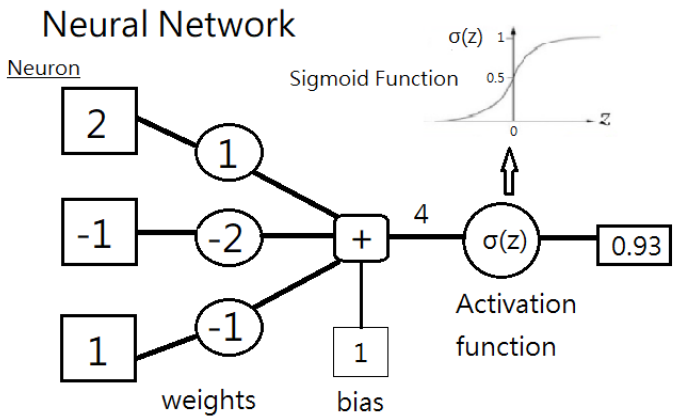


(圖 1)：Neural Network Structure

資料來源: Deep Learning, book by Ian Goodfellow, Yoshua Bengio, and Aaron Courville (2019)

4.1.3 DNN(Deep Neural Network)

在深度學習（DL）領域中，建構神經網路（Neural Network，以下簡稱 NN）時，我們使用神經元組成的模型進行訓練。NN 的結構包括輸入層、輸出層以及中間層，其中輸入層和輸出層的節點數目通常是固定的，而中間層的節點數目可以自由指定。每個神經元由輸入變數、權重（Weights）、偏差（Bias）以及激勵函數（Activation function）組成，它將輸入值轉換為一個輸出值。如(圖 2)所示，我們可以通過調整神經元和層的組成來設計不同結構的神經網路。



(圖 2)：Neural Network Computing Process

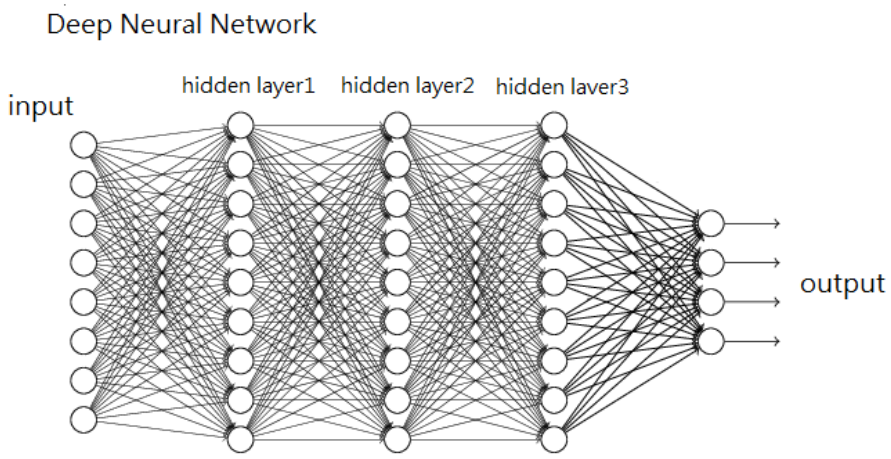
資料來源:Deep Learning, book by Ian Goodfellow, Yoshua Bengio, and Aaron Courville (2019)

在 DL 的簡述中，我們探討了具有多個且多層神經元的神經網路(NN)。其中，所謂的多層通常指的是隱藏層(Hidden Layer)。DL 的設計涉及多個層次，包括決定神經網路的層數、每層的節點數(神經元)、節點之間的連接方式、激勵函數的選擇以及優化器的選用等，這些統稱為網路架構。我們可以視 DL 的設計為神經網路的本質特徵。而權重和偏差則是透過大量的資料自動學習獲得的，可以看作是神經網路訓練後的結果。如(式 5)(式 6)。

$$Y = \sum(weight * input) + bias \quad \text{(式 5)}$$

$$Z = x_1 w_1 + x_2 w_2 + \ldots\ldots + x_n w_n + b * 1 \quad \text{(式 6)}$$

類神經網絡模擬人類腦細胞的運作原理，以實現「判斷」功能。人類腦細胞中存在著眾多的神經元(Neurons)，這些神經元通過突觸相互連接，接收外部訊號後轉換成輸出，再傳遞到下一個神經元。每個神經元具有不同的轉換能力，透過這樣的訊號傳遞與轉化，形成了人類的思考和判斷能力。類神經網路通過模仿這種人腦的運作方式來實現相似的效果。單層轉換的效果有限，因此我們會添加其他層來增加系統的穩定性，這些額外的層被稱為隱藏層，並且所有輸入層和隱藏層之間的神經元是完全相連的。如(圖 3)示。



(圖 3)：Deep Neural Network(DNN) Diagram

資料來源:Deep Learning, book by Ian Goodfellow, Yoshua Bengio, and Aaron Courville (2019)

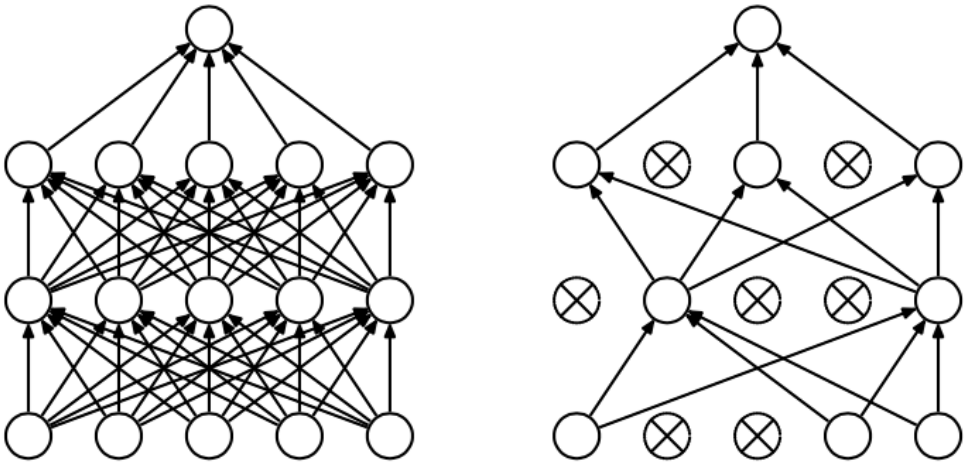
理論上，深度神經網路(DNN)的中間隱藏層越深越有利。DNN 有時也被稱為多層感知機(Multi-Layer perceptron,MLP)，其中各層之間均為全連接，即第 i 層的每個神經元與第 i+1 層的每個神經元相連。儘管 DNN 看起來複雜，但從局部模型的角度來看，與感知器類似，即構成一個線性關係$z = \sum w_i x_i + by$，再加上一個激勵函數 σ(z)。

隨著資料集的維度大小和資料量的不同，除了激勵函數外，學習率(Learning rate)和中間神經網路層的神經元個數也會對模型產生影響。我們定義了理想的正確分類得分與當前權重所計算得分之間的差距為損失函數(loss)。因此，訓練深度神經網路的目標是找到一組權重，使得對於給定較大規模數據集的 loss 最小化。梯度值告訴我們權重應如何調整以減少損失，這一過程是通過重複迭代進行的。

4.1.4 Drop Out 層

在機器學習中，若是模型的引數太多，且訓練樣本過少，就容易產生 Overfitt 的現象，許多機器學習的模型都會遇到該問題，為了解決該問題就會在模型當中加入 Drop Out 以有效的緩解過擬和的發生。

DropOut 的概念首先由 Hinton 在其論文中《Improving neural networks by preventing co-adaptation of feature detectors》[44]提出。Drop Out 可以做為訓練的一種 trick 供使用，在每個訓練中都會忽略掉一半的特徵檢測，讓每個特徵檢測器不會過於互相依賴，以至於整個模型不會過於依賴某些區域的特徵。



(圖 4)：Drop Out model

資料來源: Deep Learning, book by Ian Goodfellow, Yoshua Bengio, and Aaron Courville (2019)

4.1.5 激勵函數(Activation Function)

在類神經網路中若不使用激勵函數，則每一層的輸出將僅是以上一層輸入的線性組合(即矩陣相乘)，這導致輸出仍然呈現線性關係，深度類神經網路將失去意義。激勵函數通常是非線性的，其主要功能是模擬神經元間的信號傳遞過程，即定義神經元如何根據其他神經元的輸入改變自身的激勵程度。由於激勵函數是

非線性的，因此它的加入使得神經網路能夠處理更加複雜的非線性問題。

常看到實驗所選擇的激勵函數為 Leaky_ReLU，如(式 7)

$$f(x) = max(0.01x, x) \quad (式\ 7)$$

最近年來，ReLu 成為最常用的激勵函數之一，因其具有以下特點：解決梯度爆炸問題、計算速度快、收斂速度迅速等。在深度學習領域，ReLu 激勵函數已成為主流。其主要優勢在於其分段線性特性能有效克服梯度消失的問題。為了解決 Dead ReLU Problem，人們提出了將 ReLU 的前半段設為 $0.01x$倍而非 0 的方法。Leaky_ReLU 擁有所有 ReLU 的優點，並避免了 Dead ReLU 的問題。

4.2 支持向量機(Support Vector Machine)

SVM 是一種監督式學習演算法，旨在分析數據並進行模型識別。這一方法源於統計學習理論(Statistical Learning Theory)，最初以簡易向量分類器(Simple Vector Classifiers)的形式出現，逐漸演化為超平面分類器，最終發展成為支持向量分類器(Support Vector Machine)。SVM 廣泛應用於多個領域，包括手寫辨識、影像識別、文字分類以及生物科技等各種相關的分類問題。

支持向量機主要是使用超平面(Separating Hyperplane)[45]來分割不同類別的資料，這些資料可能有兩個或多個類別，這種方法可以用在資料探勘分類處理上的問題。在基本分類相關問題中，我們將x_i定義為一個向量，代表著某筆 N 維的樣式(Pattern)或屬性(Attribute)，如(式 8)

$$x_i \epsilon R^N, i = 1,2,3,\dots,m \quad (式\ 8)$$

y_i稱為標籤，通常用$\{\pm 1\}$來表示兩類問題的分類標籤，+1和-1分別代表不同的類別。如(式9)

$$y_i \epsilon \{\pm 1\}, i = 1,2,3,\dots,m \quad (式\ 9)$$

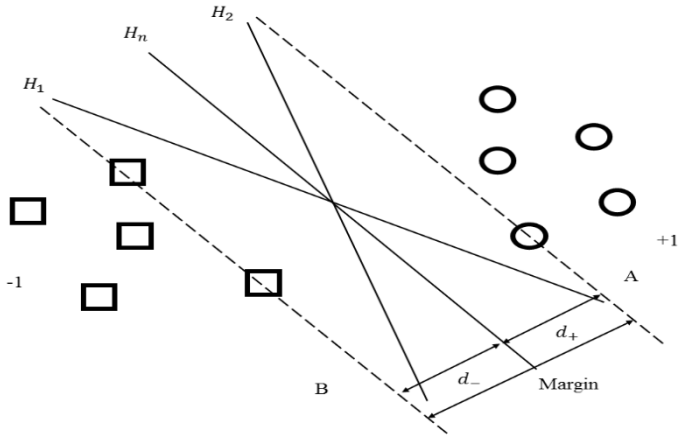
SVM依據處理問題的不同，通常可以又分成線性支持向量機以及非線性支持向量機兩類。

線性的支持向量機分類主要目標是想找到一個區分超平面(Separating Hyperplane)，它能夠把資料分隔成最大邊界(Margin)，如(圖2)，A和B分別代表不

同的類別，分別用+1以及-1表示，d則是不同類別的資料和超平面的最短距離，此類資料必須要符合以下的(式10)

$$(x_1, y_1), \ldots, (x_i, y_i), x_i \epsilon R^d, y_i \epsilon \{-1,1\} \quad (式10)$$

假設有個超平面可將這兩類的資料分開，我們稱之為區分超平面，任何x在這個區分超平面上，都須符合$w \cdot x + z = 0$，其中w為超平面的法向量，z是偏移量。我們可以將$f(x) = w \cdot x + z = 0$稱為決定函數，若有一筆資料需要輸入時，就能使用這個函數來判斷其屬於A類還是B類。支持向量機的目標就是希望能夠在不同的類別中找到一個最大邊界的區分超平面。由(圖5)我們發現除了H_n之外，還有H_1以及H_2可以用來區分這兩類，而H_n則是和A以及B兩類問題的邊界距離最大。



（圖 5）：The Best Hyperplane To Distinguish Between Two Types of Problems

資料來源: Corinna Cortes & Vladimir Vapnik “Support-vector networks” Machine Learning volume 20, pages273–297(1995)

首先我們定義d_+和d_-分別表示為+1以及-1兩種類別的距離區分超平面的最短距離，必須滿足以下公式：

$$x_i \cdot w + z \geq +1, y_i = +1 \quad (式11)$$

$$x_i \cdot w + z \leq -1, y_i = -1 \quad (式12)$$

將(式11)以及(式12)結合成如下(式13)：

$$y_i(x_i \cdot w + z) - 1 \geq 0, \forall i \quad (式13)$$

由(式11)以及(式12)分別得到$w \cdot x + z = 0$的距離為$\frac{1}{\|w\|}$，因此$d_+ = d_- = \frac{1}{\|w\|}$，邊界為$\frac{2}{\|w\|}$。我們找出最大邊界的區分超平面就如同在符合(式12)之下所求的$\|w\|^2$最小值。在(式13)中，如果有符合的x_i可以使等號成立，就可以將x_i稱作支持向量。為了方便計算，在(式13)求$\|w\|^2$最小值我們可以使用拉格朗日(Lagrange)最佳化的問題來處理，公式表示如下：

$$L_p \equiv L(w,a) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} a_i[y_i(w \cdot x_i) + z] - \sum_{i=1}^{m} a_i \quad \text{(式14)}$$

其中拉格朗日系數$a_i, i = 1,2,3,\ldots,m$對應到(式14)中的每一個不等式，而且$a_i > 0$。我們就可以將原先的問題轉化成求L_p最小值的問題而且限制式為$a_i > 0$。

用拉格朗日最佳化對偶問題(Lagrange Dual Optimization Problem)來解決，先對(式13)的w和z做偏微分：

$$\frac{\partial}{\partial w} L_p = 0 \text{ ，} w = \sum_{i=1}^{m} a_i y_i x_i \quad \text{(式15)}$$

$$\frac{\partial}{\partial b} L_p = 0 \text{ ，} \sum_{i=1}^{m} a_i y_i = 0 \quad \text{(式16)}$$

將(式15)以及(式16)帶入(式14)，經過整理後可以得到(式17)，我們將此函式命名為L_D，經由求對偶問題L_D的最大值取代求L_p最小值的問題。

$$\begin{gathered} L_D = \textstyle\sum_{i=1}^{m} a_i - \frac{1}{2}\sum_{i,j=0}^{m} a_i a_j y_i y_j (x_i \cdot x_j) \\ Subject\ to: a_i \geq 0 \quad i = 1,\ldots,m \quad and \quad \textstyle\sum_{i=1}^{m} a_i y_i = 0 \end{gathered} \quad \text{(式17)}$$

根據Karush Kuhn-Tucker(KKT)，可以得到(式18)

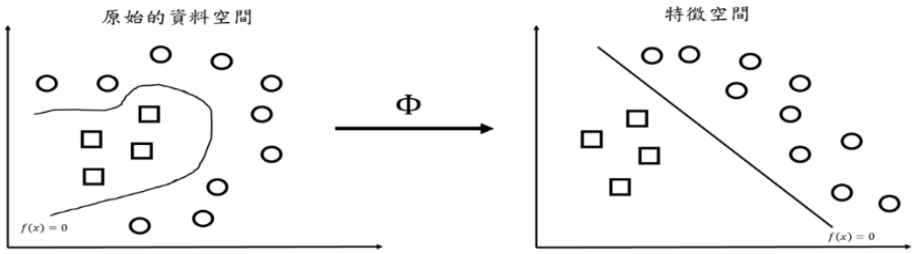
$$\sum_{i=1}^{m} a_i^* [y_i^* (< w_i^* \cdot x_i > + z^* - 1)] = 0 \quad \text{(式18)}$$

假設有x_i符合上式，那當中的x_i就會是最趨近最佳化區分超平面之向量，換句話說若有一個x_i的$a_i^* \geq 0$，我們就能稱x_i為支持向量。找出支持向量之後，即可以找出最大邊界。最後能夠取得一個分類函數如下：

$$f(x) = sgn\left(\sum_{i=1}^{m} y_i a_i \cdot \left(x_i \cdot x_j\right) + z\right) \quad \text{(式19)}$$

當$f(x) > 0$代表該筆資料與+1屬於同一類，反之則代表另外一類。

在現實情況下，並非所有資料皆能透過線性區分超平面得以分類。尤其是當面對非線性的資料集時，尋找線性區分超平面便無法滿足需求。為了解決這一問題，Boser和Vapnik 等學者提出了一種解決方案，即透過非線性映射函數Φ，將原始資料轉換至另一高維度的特徵空間中，然後在該特徵空間中進行分類，如(圖6)



（圖 6）：Original Data Is Transformed Into The Feature Space By Φ

資料來源：Corinna Cortes & Vladimir Vapnik “Support-vector networks” Machine Learning volume 20, pages273–297(1995)

在最佳化對偶問題的時候(式17)，會影響最後結果的$\left(x_i \cdot x_j\right)$，若把資料透過Φ轉化到特徵空間，那麼影響到結果的就會是$\Phi(x_i) \cdot \Phi(x_j)$，而$\Phi(x_i)$以及$\Phi(x_j)$的內積，然後可以用核心函數(Kernel Function)替代，所以只要特徵空間的內積值能夠由核心函數算出資料，就不用把資料直接映射到特徵空間，如(式19)

$$k\left(x_i, x_j\right) := \left(\Phi(x_i) \cdot \Phi\left(x_j\right)\right) \quad \text{(式20)}$$

所以非線性的支持向量機所處理的最佳化問題可以改寫為：

$$L_D = \sum_{i=1}^{m} a_i - \frac{1}{2}\sum_{i,j=0}^{m} a_i a_j y_i y_j k(x_i \cdot x_j)$$
$$Subject\ to: 0 \le a_i \le C \quad i = 1,2,3 \ldots, m \quad and \quad \sum_{i=1}^{m} a_i y_i = 0 \qquad (式21)$$

底下為常見的核心函數，有四種分別為有線性(Linear)、多項式(Polynomial)、放射(Radial Basis Function, RBF)和S型(Sigmoid)。每種核心函數都有參數可以根據不一樣的需求來做調整。

Linear kernel： $k(x_i, x_j) = x_i \cdot x_j^T$

Polynomial kernel： $k(x_i, x_j) = (1 + x_i \cdot x_j)^d$

Radial Basis Function kernel： $k(x_i, x_j) = exp(-\gamma \left\| x_i - x_j \right\|^2)$

Sigmoid kernel： $k(x_i, x_j) = tanh\,(kx_i \cdot x_j - \delta)$

透過適合之kernel function，我們可以不必經由映射函數轉換向量到高維空間，且能更快找到適合之超平面來對資料做分類。

4.3 K-近鄰演算法(k-nearest neighbors algorithm)

K-近鄰演算法(k-nearest neighbors algorithm, K-NN)[46]是一種演算法，在監督式(Supervised Learning)的機器學習模式下可被用來解決分類(Classification)和回歸(Regression)等問題。在分類(Classification)和回歸(Regression)問題中，將輸入找尋 K 個最接近的訓練樣本，輸出是一個測試資料被分類的族群。一個測試資料的分類是由其 K 個鄰居所屬族群的多數決而定，其中 K 為正整數。若 K 為 1 則測試資料直接由最近的一個鄰居決定其族群。

在訓練階段，K-近鄰演算法是一種監督式學習方法。在此階段，預測模型僅使用訓練集中每一筆資料的特徵向量和對應的分類標籤進行訓練。K 個鄰居將根據這些特徵向量從訓練集中進行挑選。至於分類階段，K 的數目由使用者自訂。未標記的資料將被歸類為與其最接近的 K 個鄰居所屬最多的類別。

一般情況下，歐基里德距離(Euclidean distance)(式 27)常被作為距離度量，但僅適用於連續分布(Continuous distribution)的資料。不同的距離度量方法會影響K-近鄰演算法分類的精度，而在 Li-Yu Hu 的研究證明中距離函數的選擇對 K-NN 分類器的精確度具有重要影響。K-近鄰演算法以其簡單易懂、易於實現、無需估計參數和無需事先訓練樣本等優點聞名，特別適合多分類問題(multi-class)。然而，

當類別的樣本數量不平衡時，K-NN 演算法存在一些缺陷，例如可能導致類別分布偏斜，一個類的樣本數量如果很多，而其他類樣本數量很少時，有可能導致當一個新樣本要輸入時，該樣本的 K 個鄰居中大數量類別的樣本占多數，另一個不足之處是計算量較大，因為對每一個待分類的資料都要計算它到全體已知樣本的距離，才能求得它的 K 個最近鄰點。近期的研究表明，提出了一些新的方法，例如 Zhenfeng Lei[47]的研究，使得 K-近鄰演算法在預測蛋白質亞細胞定位(sub-cellular localization)可用低維空間表達高維數據，實現更好的分類效果。

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (式\ 22)$$

4.4 實驗資料

本論文使用了 STRING 資料庫提供的綠膿桿菌(Pseudomonas Aeruginosa)資料。首先，根據前幾章節所述的方法，我們刪除了胺基酸序列長度小於 50 的資料，並利用 cdhit 去除相似度大於等於 40%的蛋白質對，接著，我們從訓練集(train set)和測試集(test set)中生成了一定數量的非交互作用對，是採用 Ushuffle 和 LeftRight 方法來生成這些非交互作用對。本論文的交互作用蛋白質對(Positive data)在訓練集有 11289 筆，在測試集則有 2995 筆，另外，非交互作用蛋白質對(Negative data)在訓練集中分別 Ushuffle-K1 有 11289 筆和 LeftRight 的 22578 筆，測試集的則分別是 Ushuffle-K1 的 2995 筆和 LeftRight 的 5990 筆。

蛋白質描述方法的部分，我們使用了協方差(AC)作為特徵提取的方法，並設定了 lags 值為 10、20、30、40、50。同時，我們採用了簡化胺基酸(RAAA)方法，搭配不同的肽鍵長度，包括了 RAAA1_lt2、RAAA2_lt2、RAAA2_lt3、RAAA3_lt3 等。將兩條蛋白質序列的胺基酸特徵向量放入分類器中進行訓練和預測。以 RAAA1-lt2 為例，一條胺基酸序列的維度為 400，兩條則為 800 維；RAAA2-lt2 一條胺基酸序列的維度為 169，兩條則為 338(維)，依此類推。

我們採用了三種不同的分類器來進行交互作用實驗的預測，包括 KNN、SVM 以及深度神經網路(DNN)，DNN 中分為加入 DropOut 層，以及無 DropOut 層的 DNN_ND。

對於 KNN 模型，由於其屬於較簡單的分類器，準確性也相對較低。我們通過測試不同的 K 值(1~51)，並在結果表格中列出最佳的準確率作為 K 值的結果。另外，在神經網路系統中，我們選擇使用激勵函數 ReLu，這是一種近年來最常用的激勵函數，其特點包括計算速度快、收斂速度快以及能夠解決梯度消失的問

題。作為優化器，我們採用了 AdamOptimizer [48]，它結合了 AdaGrad 和 RMSProp 兩種優化演算法的優點，具有對記憶體使用少、自動調節學習率以及參數更新不受梯度變換影響等特點。

在模型建立方面，我們利用了 Keras 程式庫提供的 TensorFlow 函式庫，它提供了完整的深度學習框架，同時也能夠方便地調用 Dropout。本篇論文的神經網路模型都使用了 Keras 提供的回調函數，包括 ModelCheckpoint 和 EarlyStopping。ModelCheckpoint 負責在訓練過程中保存最佳模型，而 EarlyStopping 則根據 Patience 參數，在訓練過程中當準確率不再提高時停止訓練，以節省時間和資源。

第五章 實驗與分析

5.1 評估預測表現的方法跟標準

在機器學習中常用交叉驗證法(Cross Validation)來評估方法，而交叉驗證又可分為幾種，裡面有兩種較常被使用分別為 K-fold Cross Validation 以及 Leave One Out Cross Validation(LOOCV)，以下將分別做介紹。

在模式識別(Pattern Recognition)和機器學習(Machine Learning)的研究中，通常會將資料集分為訓練集(Train Set)和測試集(Test Set)兩個子集。訓練集的主要目的是用來建立模型(Model)，而測試集則用於評估模型對未知資料進行預測時的準確度，這也被稱為模型的泛化能力(Generalization Ability)。除了訓練集和測試集外，有時候還會使用驗證集(validation set)，用於調節模型的超參數(hyperparameters)以及評估不同模型的性能。確保有良好的訓練、驗證和測試集分割是確保模型在真實世界中表現良好的重要步驟。

在統計學上，K-fold Cross Validation 是一種非常實用的方法，它將數據訓練樣本切割成較小的子集。這對於測試樣本的取樣和收集不容易達成的時候非常重要，因為它可以多次實驗來驗證評估模型的性能。交叉驗證法將訓練樣本隨機地分成 k 份子集，每一份都是大小相同且彼此互不相交的子集。然後，利用其中的 k-1 份作為訓練集，剩下的一個子集作為測試集。重複進行 k 次，每一個子集都有機會進行測試，最後計算其準確率的平均值以獲得模型的性能評估。

相比於K-fold Cross Validation，在不考慮時間因素和計算量的情況下LOOCV在預測準確率上更具優勢，若是在效率和效能上來看，還是比 K-fold Cross Validation 差了些。LOOCV 是交叉驗證法的一種，當對蛋白質間交互作用的資料集進行評估時，每一筆資料都被單獨當做測試集，而其他交互作用資料則用於訓練集。若共有 N 筆蛋白質對資料，則需執行 N 次分類器的運算。依序對所有蛋白質資料進行測試後，可以統計正確預測蛋白質對的次數 m，進而得到整體的預測準確率 m/N。

5.1.1 預測表現的評斷標準

在機器學習的分類預測問題中，評估預測表現的方法具有多樣性。一般而言，常用的評斷方法之一是準確率(Accuracy)，其計算方式為將分類器正確分類的樣本數與總樣本數進行比較。準確率的優勢在於其能夠較為容易地評估整體預測系統的表現，且結果以百分比形式呈現，範圍在 0%至 100%之間。然而，當資料量

過大時，使用準確率可能不容易顯示出差異。

除了準確率外，在本篇論文中，我們還採用了其他指標來評估分類器的預測表現，其中包括精確率(Precision)、靈敏度(Sensitivity)，精確率主要是放在預測為 Positive 的樣本中實際上有多少為 Positive，在本論文會使用上述三種判斷方式來判斷系統性能。這些指標能夠提供更全面的評估，有助於了解模型在不同方面的表現，從而更深入地評價其預測能力。

$$Sensitivity = \frac{TP}{TP+FN} \quad (式\ 23)$$

$$Precision = \frac{TP}{TP+FP} \quad (式\ 24)$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (式\ 25)$$

(式 23)到(式 25)中出現的各個符號說明如下： TP(True Positives)為預測例證屬於此類，並且預測正確的數量。FP (False Positives)為預測例證屬於此類，但預測錯誤的數量。TN(True Negatives)為預測例證不屬於此類，並且預測正確的數量。FN(False Negatives)為預測例證不屬於此類，但預測錯誤的數量。另外，這四個符號可以組合成我們熟知的混淆矩陣(Confusion Matrix)，可以透過這個矩陣更好的看到每個模型性能和評估分析上的表現。

5.1.2 實驗結果

以下為使用 Ushuffle 偽造蛋白質不交互作用對，以及使用簡化胺基酸搭配選取不同長度肽的蛋白質描述法的預測結果，Ushuffle 使用 k-let 為參數分為 K1。

(表 3) Ushuffle-seed0-K1-287_peptide_nm1_RAAA1_lt2

	TP	FP	TN	FN	Sensitivity	Precision	Test ACC
KNN					0.850	0.851	0.850
SVM	2617	399	2596	378	0.874	0.868	0.870
DNN	2719	105	2890	276	0.908	**0.963**	0.936
DNN_ND	2862	153	2842	133	**0.956**	0.949	**0.952**

(表 4) Ushuffle-seed0-K1-287_peptide_nm1_RAAA2_lt2

	TP	FP	TN	FN	Sensitivity	Precision	Test ACC
KNN					0.812	0.818	0.812
SVM	2624	370	2625	371	0.876	0.876	0.876
DNN	2721	193	2802	274	**0.909**	**0.934**	**0.922**
DNN_ND	2682	221	2774	313	0.895	0.924	0.911

(表 5) Ushuffle-seed0-K1-287_peptide_nm1_RAAA2_lt3

	TP	FP	TN	FN	Sensitivity	Precision	Test ACC
KNN					0.827	0.834	0.827
SVM	2719	215	2780	276	0.908	0.927	0.918
DNN	2854	127	2868	141	0.953	**0.957**	0.955
DNN_ND	2864	134	2861	131	**0.956**	0.955	**0.956**

(表 6) Ushuffle-seed0-K1-287_peptide_nm1_RAAA3_lt3

	TP	FP	TN	FN	Sensitivity	Precision	Test ACC
KNN					0.795	0.814	0.795
SVM	2791	236	2759	204	0.932	0.922	0.927
DNN	2858	164	2831	137	**0.954**	**0.946**	**0.950**
DNN_ND	2806	190	2805	189	0.937	0.937	0.937

(表 3) RAAA1-lt2，DNN_ND 表現比較好一點，Accuracy 當中最高的是 DNN_ND 可以到 95.2%，Precision 最高的是 DNN 可以到 96.3%，Sensitivity 最高的是 DNN_ND 的 95.6%。

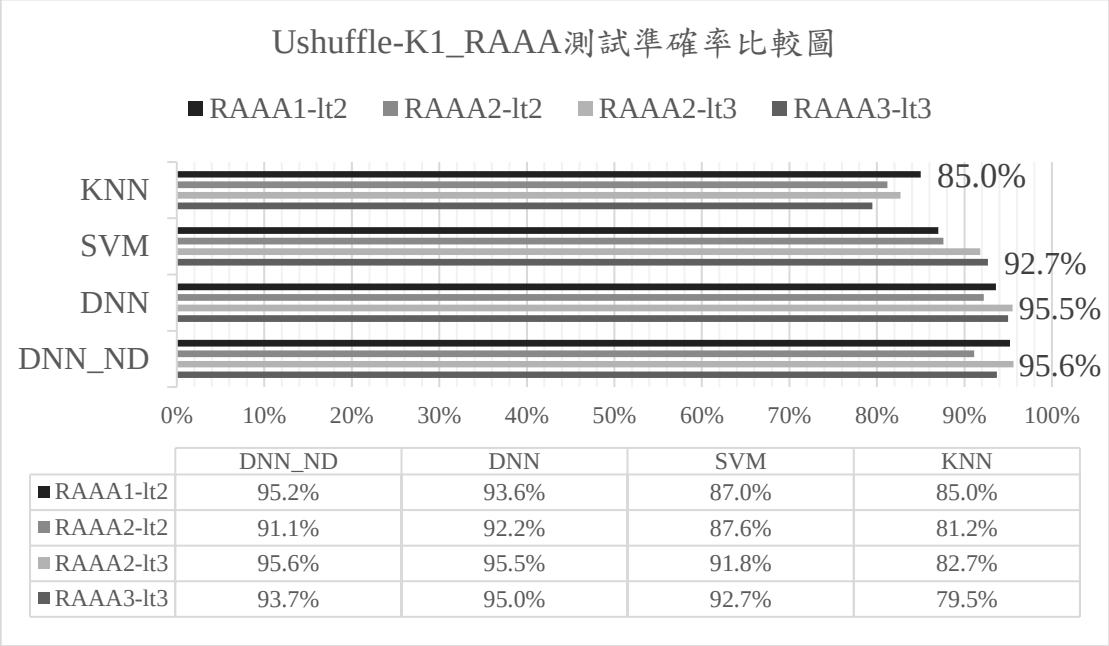
(表 4) RAAA2-lt2，DNN 整體表現最好，Accuracy 當中最高的是 DNN 可以到 92.2%，Precision 最高的是 DNN 可以到 93.4%，Sensitivity 最高的也是 DNN 的 90.9%。

(表 5) RAAA2-lt3，DNN_ND 和 DNN 的整體表現一樣好，Accuracy 當中最高的是 DNN_ND 可以到 95.6%，而 DNN 也可以達到 95.5%，Precision 最高的是 DNN

可以到 95.7%，而 DNN_ND 也可以達到 95.5%，Sensitivity 最高的是 DNN_ND 的 95.6%，而 DNN 也可以達到 95.3%。
(表 6) RAAA3-lt3，DNN 表現最好，Accuracy 當中最高的是 DNN 可以到 95%，Precision 最高的是 DNN 可以到 94.6%，Sensitivity 最高的也是 DNN 的 95.4%。

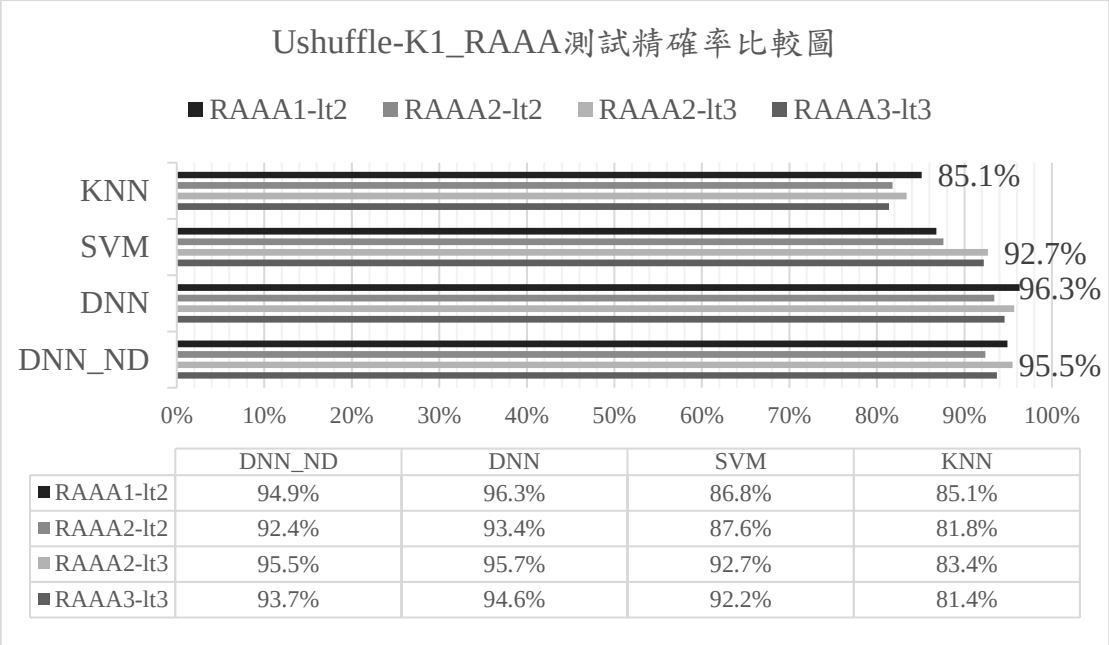
在(表 3)~(表 6) 中顯示 RAAA1-lt2、RAAA2-lt2、RAAA2-lt3、RAAA3-lt3，藉由 Ushuffle-K1 偽造蛋白質不交互作用對，對於幾種分類器的測試結果。

KNN 最高測試準確率、精確率和靈敏度都是在 RAAA1-lt2 的時候，分別為 85%、85.1%和 85%。SVM 最高測試準確率為 RAAA3-lt3 的 92.7%，最高測試精確率為 RAAA2-lt3 的 92.7%，而最高測試靈敏度為 RAAA3-lt3 的 93.2%。DNN 最高測試準確率為 RAAA2-lt3 的 95.5%，最高測試精確率為 RAAA1-lt2 的 96.3%。而最高測試靈敏度為 RAAA3-lt3 的 95.4%。DNN_ND 最高測試準確率為 RAAA2-lt3 的 95.6%，最高測試精確率為 RAAA2-lt3 的 95.5%，而最高測試靈敏度為 RAAA1-lt2 和 RAAA2-lt3 的 95.6%。整體來看，DNN 和 DNN_ND 的效果都是比較好的，如(圖 7) (圖 8) (圖 9)。



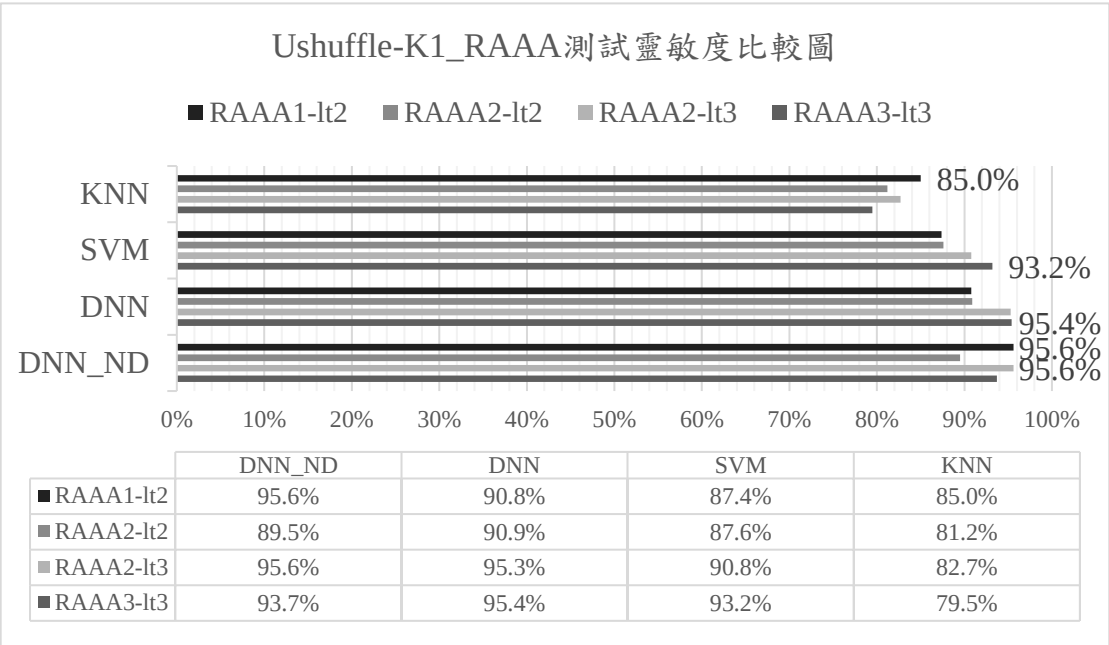
	DNN_ND	DNN	SVM	KNN
RAAA1-lt2	95.2%	93.6%	87.0%	85.0%
RAAA2-lt2	91.1%	92.2%	87.6%	81.2%
RAAA2-lt3	95.6%	95.5%	91.8%	82.7%
RAAA3-lt3	93.7%	95.0%	92.7%	79.5%

(圖 7) : Uushuffle-K1_RAAA 測試準確率比較圖



	DNN_ND	DNN	SVM	KNN
RAAA1-lt2	94.9%	96.3%	86.8%	85.1%
RAAA2-lt2	92.4%	93.4%	87.6%	81.8%
RAAA2-lt3	95.5%	95.7%	92.7%	83.4%
RAAA3-lt3	93.7%	94.6%	92.2%	81.4%

(圖 8) : Uushuffle-K1_RAAA 測試精確率比較圖



	DNN_ND	DNN	SVM	KNN
RAAA1-lt2	95.6%	90.8%	87.4%	85.0%
RAAA2-lt2	89.5%	90.9%	87.6%	81.2%
RAAA2-lt3	95.6%	95.3%	90.8%	82.7%
RAAA3-lt3	93.7%	95.4%	93.2%	79.5%

(圖 9) : Uushuffle-K1_RAAA 測試靈敏度比較圖

以下為使用 LeftRight 偽造蛋白質不交互作用對，以及使用簡化胺基酸搭配選取不同長度肽的蛋白質描述法的預測結果。

(表 7) LeftRight-seed0-287_peptide_nm1_RAAA1_lt2

	TP	FP	TN	FN	Sensitivity	Precision	Test ACC
KNN					0.603	**0.609**	0.659
SVM	1346	946	5044	1649	0.449	0.587	0.711
DNN	1838	1282	4708	1157	**0.614**	0.589	**0.729**
DNN_ND	1611	1219	4771	1384	0.538	0.569	0.710

(表 8) LeftRight-seed0-287_peptide_nm1_RAAA2_lt2

	TP	FP	TN	FN	Sensitivity	Precision	Test ACC
KNN					**0.608**	**0.614**	0.663
SVM	912	713	5277	2083	0.305	0.561	0.689
DNN	1753	1399	4591	1242	0.585	0.556	**0.706**
DNN_ND	1224	1070	4920	1771	0.409	0.534	0.684

(表 9) LeftRight-seed0-287_peptide_nm1_RAAA2_lt3

	TP	FP	TN	FN	Sensitivity	Precision	Test ACC
KNN					**0.590**	0.599	0.654
SVM	963	912	5078	2032	0.322	0.514	0.672
DNN	1655	1227	4763	1340	0.553	0.574	**0.714**
DNN_ND	1191	765	5225	1804	0.398	**0.609**	**0.714**

(表 10) LeftRight-seed0-287_peptide_nm1_RAAA3_lt3

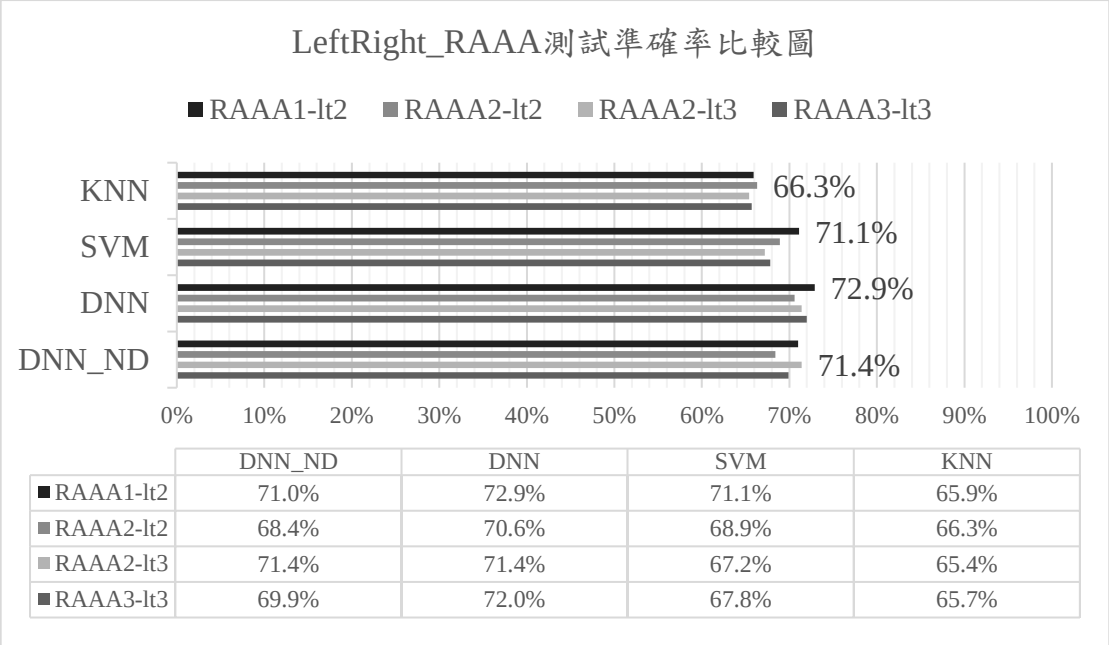
	TP	FP	TN	FN	Sensitivity	Precision	Test ACC
KNN					**0.587**	**0.601**	0.657
SVM	1078	976	5014	1917	0.360	0.525	0.678
DNN	1659	1181	4809	1336	0.554	0.584	**0.720**

DNN_ND	1476	1181	4809	1519	0.493	0.556	0.699

(表 7) RAAA1-lt2，DNN 整體表現比較好，Accuracy 當中最高的是 DNN 可以到 72.9%，Precision 最高的是 KNN 可以到 60.9%，Sensitivity 最高的是 DNN 的 61.4%。
(表 8) RAAA2-lt2，DNN 和 KNN 的整體表現都不錯，Accuracy 當中最高的是 DNN 可以到 70.6%，Precision 最高的是 KNN 可以到 61.4%，Sensitivity 最高的是 KNN 的 60.8%。
(表 9) RAAA2-lt3，DNN_ND 整體表現最好，Accuracy 當中最高的有兩個 DNN 和 DNN_ND 都可以到 71.4%，Precision 最高的是 DNN_ND 可以到 60.9%，Sensitivity 最高的是 KNN 的 59.0%。
(表 10) RAAA3-lt3，DNN 和 KNN 表現都不錯，Accuracy 當中最高的是 DNN 可以到 72%，Precision 最高的是 KNN 可以到 60.1%，Sensitivity 最高的是 KNN 的 58.7%。

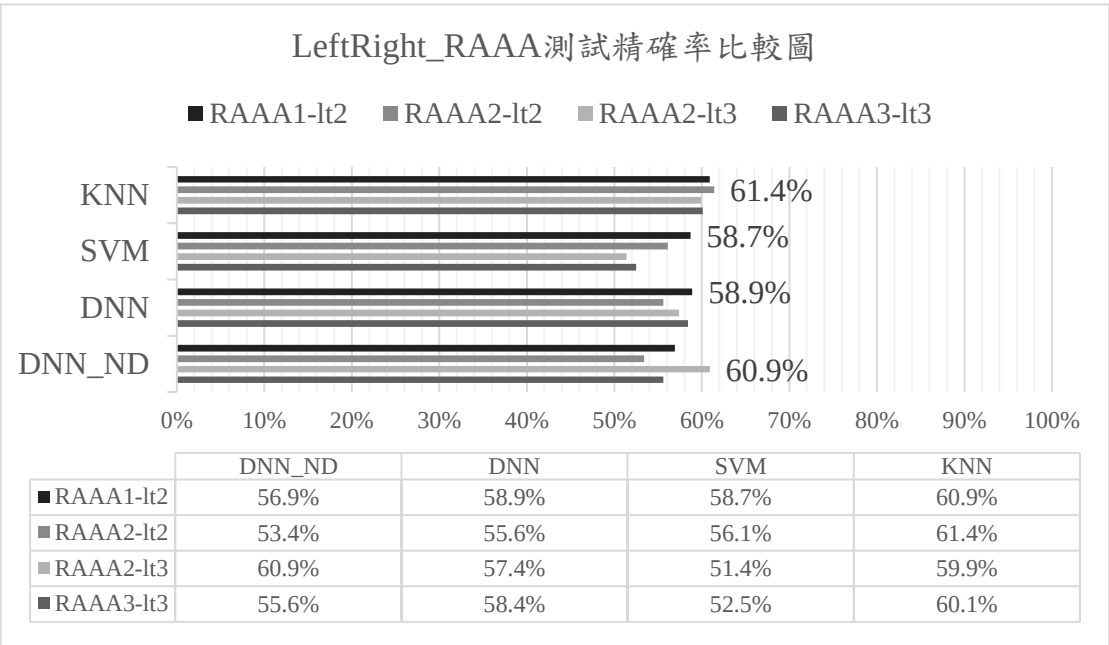
在(表 7)~(表 10) 中顯示 RAAA1-lt2、RAAA2-lt2、RAAA2-lt3、RAAA3-lt3，藉由 LeftRight 偽造蛋白質不交互作用對，對於幾種分類器的測試結果。

KNN 最高測試準確率、精確率和靈敏度都是在 RAAA2-lt2 時，分別為 66.3%、61.4%和 60.8%。SVM 最高測試準確率、精確率和靈敏度都是在 RAAA1-lt2 的時候，分別為 71.1%、58.7%和 44.9%。DNN 最高測試準確率、精確率和靈敏度都是在 RAAA1-lt2 的時候，分別為 72.9%、58.9%和 61.4%。DNN_ND 最高測試準確率為 RAAA2-lt3 的 71.4%，最高測試精確率為 RAAA2-lt3 的 60.9%，而最高靈敏度為 RAAA1-lt2 的 53.8%。整體來看，KNN 和 DNN 效果是較好的，如(圖 10) (圖 11) (圖 12)。



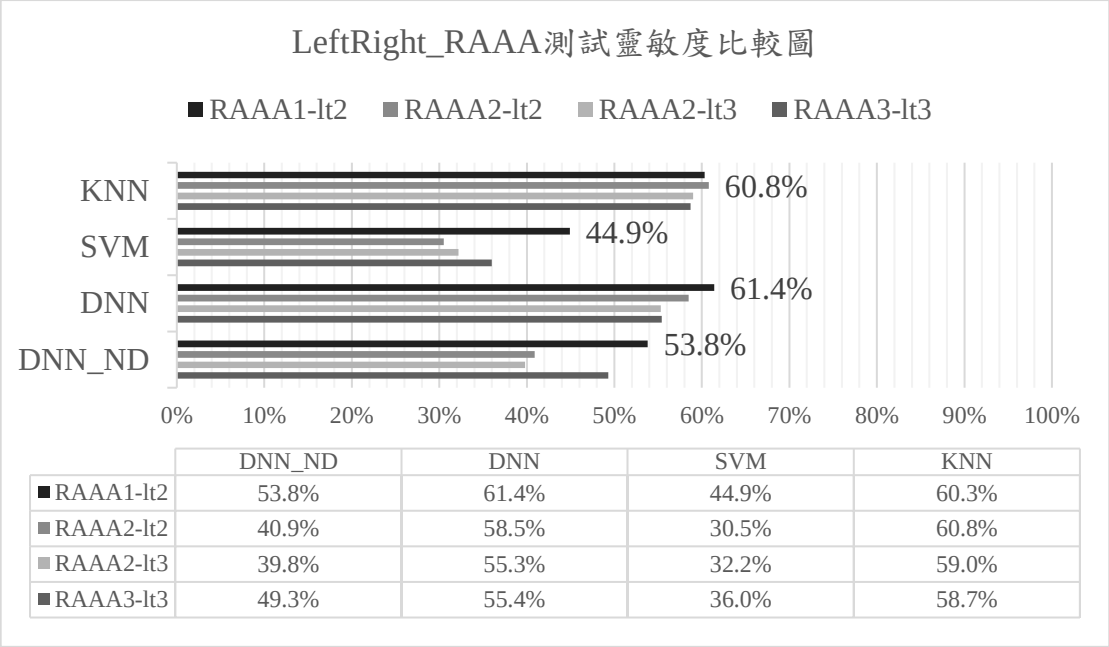
	DNN_ND	DNN	SVM	KNN
RAAA1-lt2	71.0%	72.9%	71.1%	65.9%
RAAA2-lt2	68.4%	70.6%	68.9%	66.3%
RAAA2-lt3	71.4%	71.4%	67.2%	65.4%
RAAA3-lt3	69.9%	72.0%	67.8%	65.7%

(圖 10) : LeftRight_RAAA 測試準確率比較圖



	DNN_ND	DNN	SVM	KNN
RAAA1-lt2	56.9%	58.9%	58.7%	60.9%
RAAA2-lt2	53.4%	55.6%	56.1%	61.4%
RAAA2-lt3	60.9%	57.4%	51.4%	59.9%
RAAA3-lt3	55.6%	58.4%	52.5%	60.1%

(圖 11) : LeftRight_RAAA 測試精確率比較圖



	DNN_ND	DNN	SVM	KNN
RAAA1-lt2	53.8%	61.4%	44.9%	60.3%
RAAA2-lt2	40.9%	58.5%	30.5%	60.8%
RAAA2-lt3	39.8%	55.3%	32.2%	59.0%
RAAA3-lt3	49.3%	55.4%	36.0%	58.7%

(圖 12) : LeftRight_RAAA 測試靈敏度比較圖

以下為使用 Ushuffle 偽造蛋白質不交互作用對，以及使用協方差(AC)的蛋白質描述法的預測結果，其中 lag 值分別設為 10、20、30、40、50。

(表 11) Ushuffle-seed0-K1-287_AC_10

	TP	**FP**	**TN**	**FN**	**Sensitivity**	**Precision**	**Test ACC**
KNN					0.815	0.820	0.815
SVM	2693	312	2683	302	0.899	0.896	0.898
DNN	2730	307	2688	265	**0.912**	**0.899**	**0.905**
DNN_ND	2715	330	2665	280	0.907	0.892	0.898

(表 12) Ushuffle-seed0-K1-287_AC_20

	TP	**FP**	**TN**	**FN**	**Sensitivity**	**Precision**	**Test ACC**
KNN					0.799	0.812	0.799
SVM	2609	407	2588	386	0.871	0.865	0.868
DNN	2711	343	2652	284	**0.905**	**0.888**	**0.895**

DNN_ND	2657	386	2609	338	0.887	0.873	0.879

(表 13) Ushuffle-seed0-K1-287_AC_30

	TP	**FP**	**TN**	**FN**	**Sensitivity**	**Precision**	**Test ACC**
KNN					0.804	0.809	0.804
SVM	2783	233	2762	212	**0.929**	**0.923**	**0.926**
DNN	2741	313	2682	254	0.915	0.898	0.905
DNN_ND	2628	374	2621	367	0.877	0.875	0.876

(表 14) Ushuffle-seed0-K1-287_AC_40

	TP	**FP**	**TN**	**FN**	**Sensitivity**	**Precision**	**Test ACC**
KNN					0.807	0.815	0.807
SVM	2798	254	2741	197	**0.934**	**0.917**	**0.925**
DNN	2735	286	2709	260	0.913	0.905	0.909
DNN_ND	2654	332	2663	341	0.886	0.889	0.888

(表 15) Ushuffle-seed0-K1-287_AC_50

	TP	**FP**	**TN**	**FN**	**Sensitivity**	**Precision**	**Test ACC**
KNN					0.798	0.807	0.798
SVM	2832	169	2826	163	**0.946**	**0.944**	**0.945**
DNN	2719	294	2701	276	0.908	0.902	0.905
DNN_ND	2739	345	2650	256	0.915	0.888	0.900

(表 11) lag 值為 10，DNN 整體表現比較好，Accuracy 當中最高的是 DNN 可以到 90.5%，Precision 最高的是 DNN 可以到 89.9%，但 SVM 和 DNN_ND 也分別可以到 89.6%及 89.2%，Sensitivity 最高的是 DNN 的 91.2%。

(表 12) lag 值為 20，DNN 整體表現最好，Accuracy 當中最高的是 DNN 可以到 89.5%，Precision 最高的是 DNN 可以到 88.8%，Sensitivity 最高的也是 DNN 的

90.5%。

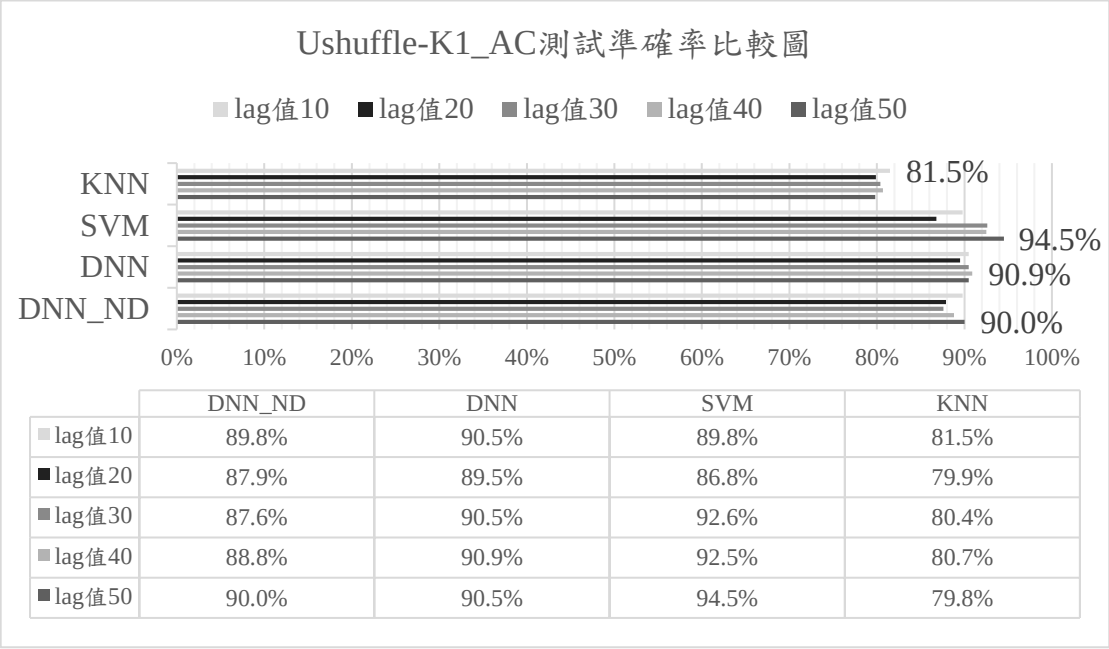
(表 13) lag 值為 30，SVM 整體表現最好，Accuracy 當中最高的是 SVM 可以到 92.6%，Precision 最高的是 SVM 可以到 92.3%，Sensitivity 最高的也是 SVM 的 92.9%。

(表 14) lag 值為 40，SVM 整體表現最好，Accuracy 當中最高的是 SVM 可以到 92.5%，Precision 最高的是 SVM 可以到 91.7%，Sensitivity 最高的也是 SVM 的 93.4%。

(表 15) lag 值為 50，SVM 整體表現最好，Accuracy 當中最高的是 SVM 可以到 94.5%，Precision 最高的是 SVM 可以到 94.4%，Sensitivity 最高的也是 SVM 的 94.6%。

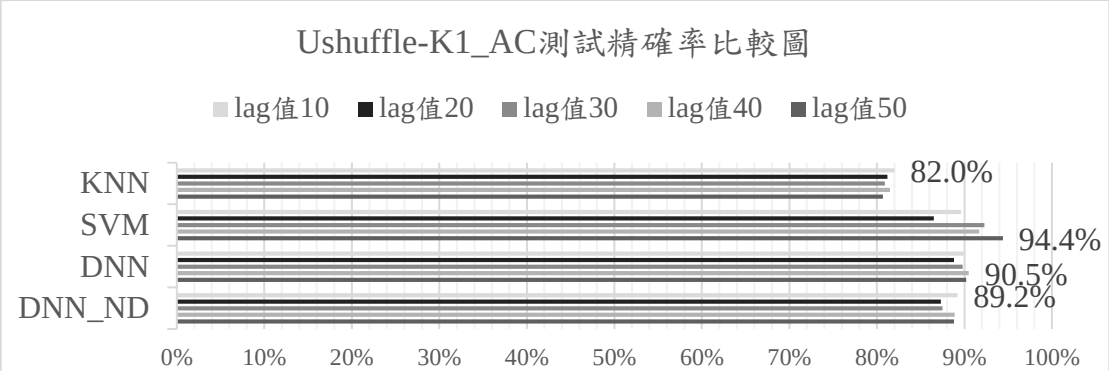
在(表 11)~(表 15) 中顯示在協方差(AC)和 lag 值為 10、20、30、40、50，藉由 Ushuffle-K1 偽造蛋白質不交互作用對，對於幾種分類器的測試結果。

KNN 最高測試準確率、精確率和靈敏度都是在 lag 值為 10 時，分別為 81.5%、82%和 81.5%。SVM 最高測試準確率、精確率和靈敏度都是在 lag 值為 50 時，分別為 94.5%、94.4%和 94.6%。DNN 最高測試準確率是在 lag 值為 40 的 90.9%，最高測試精確率是在 lag 值為 40 的 90.5%，而最高測試靈敏度是在 lag 值為 30 的 91.5%。DNN_ND 最高測試準確率是在 lag 值為 50 的 90%，最高測試精確率是在 lag 值為 10 的 89.2%，而最高測試靈敏度是在 lag 值為 50 的 91.5%。整體來看，SVM 效果是比較好的，如(圖 13) (圖 14) (圖 15)。



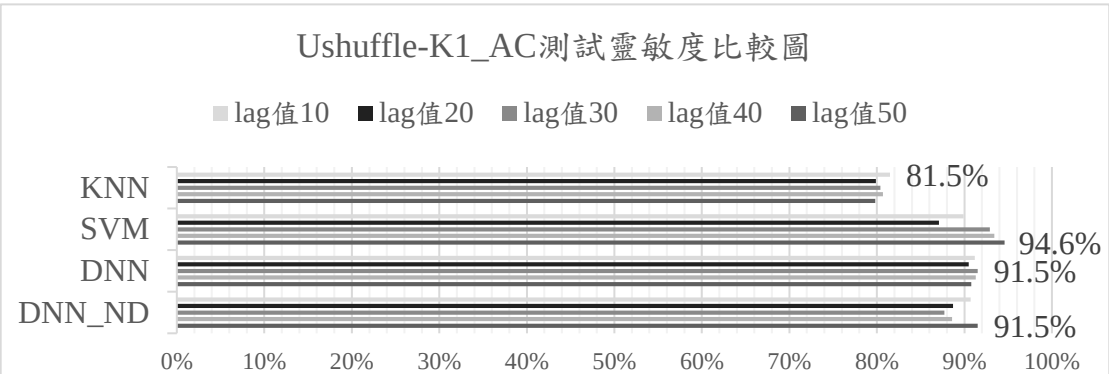
	DNN_ND	DNN	SVM	KNN
lag值10	89.8%	90.5%	89.8%	81.5%
lag值20	87.9%	89.5%	86.8%	79.9%
lag值30	87.6%	90.5%	92.6%	80.4%
lag值40	88.8%	90.9%	92.5%	80.7%
lag值50	90.0%	90.5%	94.5%	79.8%

(圖 13) : Ushuffle-K1_AC 測試準確率比較圖



	DNN_ND	DNN	SVM	KNN
lag值10	89.2%	89.9%	89.6%	82.0%
lag值20	87.3%	88.8%	86.5%	81.2%
lag值30	87.5%	89.8%	92.3%	80.9%
lag值40	88.9%	90.5%	91.7%	81.5%
lag值50	88.8%	90.2%	94.4%	80.7%

(圖 14) : Ushuffle-K1_AC 測試精確率比較圖



	DNN_ND	DNN	SVM	KNN
lag值10	90.7%	91.2%	89.9%	81.5%
lag值20	88.7%	90.5%	87.1%	79.9%
lag值30	87.7%	91.5%	92.9%	80.4%
lag值40	88.6%	91.3%	93.4%	80.7%
lag值50	91.5%	90.8%	94.6%	79.8%

(圖 15) : Ushuffle-K1_AC 測試靈敏度比較圖

以下為使用 LeftRight 偽造蛋白質不交互作用對，以及使用協方差的蛋白質描述法的預測結果，其中 lag 值分別設為 10、20、30、40、50。

(表 16) LeftRight-287_AC_10

	TP	FP	TN	FN	Sensitivity	Precision	Test ACC
KNN					**0.575**	**0.600**	0.658
SVM	772	715	5275	2223	0.258	0.519	**0.673**
DNN	1035	1299	4691	1960	0.346	0.443	0.637
DNN_ND	950	1083	4907	2045	0.317	0.467	0.652

(表 17) LeftRight-287_AC_20

	TP	FP	TN	FN	Sensitivity	Precision	Test ACC
KNN					**0.574**	**0.591**	0.653
SVM	945	799	5191	2050	0.316	0.542	**0.683**
DNN	916	1082	4908	2079	0.306	0.458	0.648
DNN_ND	1098	1230	4760	1897	0.367	0.472	0.652

(表 18) LeftRight-287_AC_30

	TP	FP	TN	FN	Sensitivity	Precision	Test ACC
KNN					**0.583**	**0.601**	0.660
SVM	887	777	5213	2108	0.296	0.533	**0.679**
DNN	1138	1308	4682	1857	0.380	0.465	0.648
DNN_ND	1196	1342	4648	1799	0.399	0.471	0.650

(表 19) LeftRight-287_AC_40

	TP	FP	TN	FN	Sensitivity	Precision	Test ACC
KNN					**0.577**	**0.600**	0.661
SVM	1140	1047	4943	1855	0.381	0.521	**0.677**
DNN	1161	1281	4709	1834	0.388	0.475	0.653

DNN_ND	1050	1107	4883	1945	0.351	0.487	0.660

(表 20) LeftRight-287_AC_50

	TP	**FP**	**TN**	**FN**	**Sensitivity**	**Precision**	**Test ACC**
KNN					**0.579**	**0.598**	0.659
SVM	1162	1108	4882	1834	0.388	0.512	**0.673**
DNN	1038	1097	4893	1957	0.347	0.486	0.660
DNN_ND	1161	1315	4675	1834	0.388	0.469	0.650

(表 16) lag 值為 10，SVM 和 KNN 整體表現差不多，Accuracy 當中最高的是 SVM 可以到 67.3，Precision 最高的是 KNN 可以到 60%，Sensitivity 最高的則是 KNN 的 57.5%。

(表 17) lag 值為 20，SVM 和 KNN 整體表現差不多，Accuracy 當中最高的是 SVM 可以到 68.3%，Precision 最高的是 KNN 可以到 59.1%，Sensitivity 最高的則是 KNN 的 57.4%。

(表 18) lag 值為 30，SVM 和 KNN 整體表現差不多，Accuracy 當中最高的是 SVM 可以到 67.9%，Precision 最高的是 KNN 可以到 60.1%，Sensitivity 最高的則是 KNN 的 58.3%。

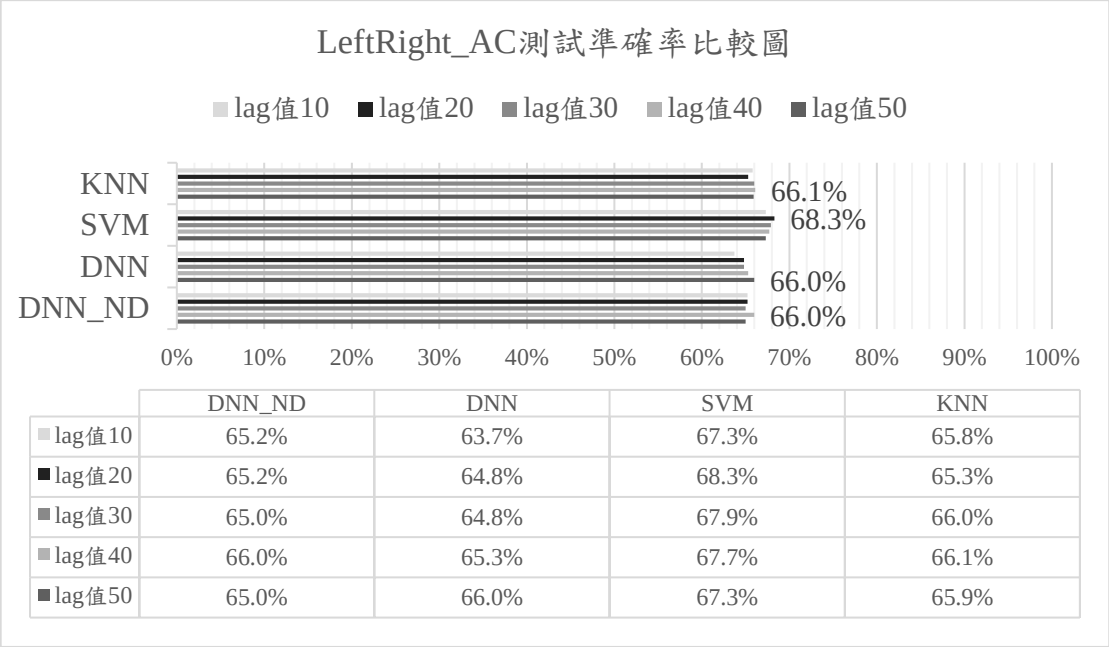
(表 19) lag 值為 40，SVM 和 KNN 整體表現差不多，Accuracy 當中最高的是 SVM 可以到 67.7%，Precision 最高的是 KNN 可以到 60%，Sensitivity 最高的則是 KNN 的 57.7%。

(表 20) lag 值為 50，SVM 和 KNN 整體表現差不多，Accuracy 當中最高的是 SVM 可以到 67.3%，Precision 最高的是 KNN 可以到 59.8% Sensitivity 最高的則是 KNN 的 57.9%。

在(表 16)~(表 20) 中顯示在協方差(AC)和 lag 值為 10、20、30、40、50，藉由 LeftRight 偽造蛋白質不交互作用對，對於幾種分類器的測試結果。

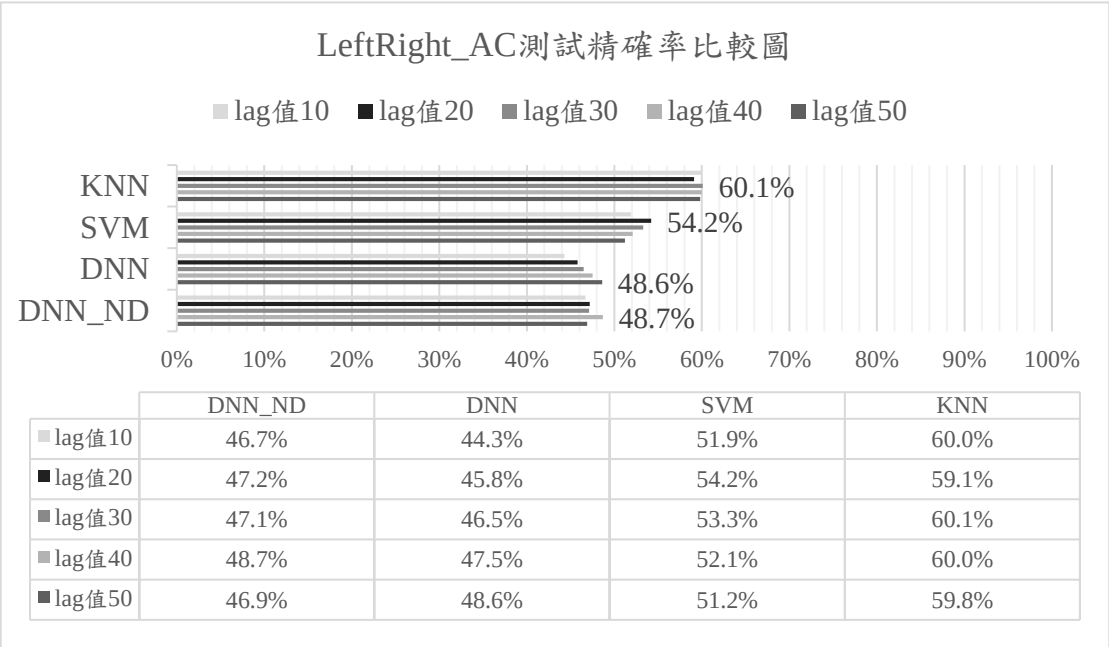
KNN 最高測試準確率是在 lag 值為 40 的 66.1%，但在 lag 值為 30 時也可以到 66%，最高測試精確率是在 lag 值為 30 的 60.1%，但在 lag 值為 10 和 40 時，也可以達到 60%，而最高測試靈敏度是在 lag 值為 30 的 58.3%。SVM 最高測試準確率是在 lag 值為 20 的 68.3%，而最高測試精確率是在 lag 值為 20 的 54.2%，而最高測試靈敏度是在 lag 值為 50 的 38.8%。DNN 最高測試準確率是在 lag 值為 50 的 66%，最高測試精確率是在 lag 值為 50 的 48.6%，而最高測試靈敏度是在 lag 值為 40 的 38.8%。DNN_ND 最高測試準確率是在 lag 值為 40 的 66%，最

高測試精確率是在 lag 值為 40 的 48.7%，而最高測試靈敏度是在 lag 值為 30 的 39.9%。整體來看，KNN 效果是比較好的，如(圖 16) (圖 17) (圖 18)。



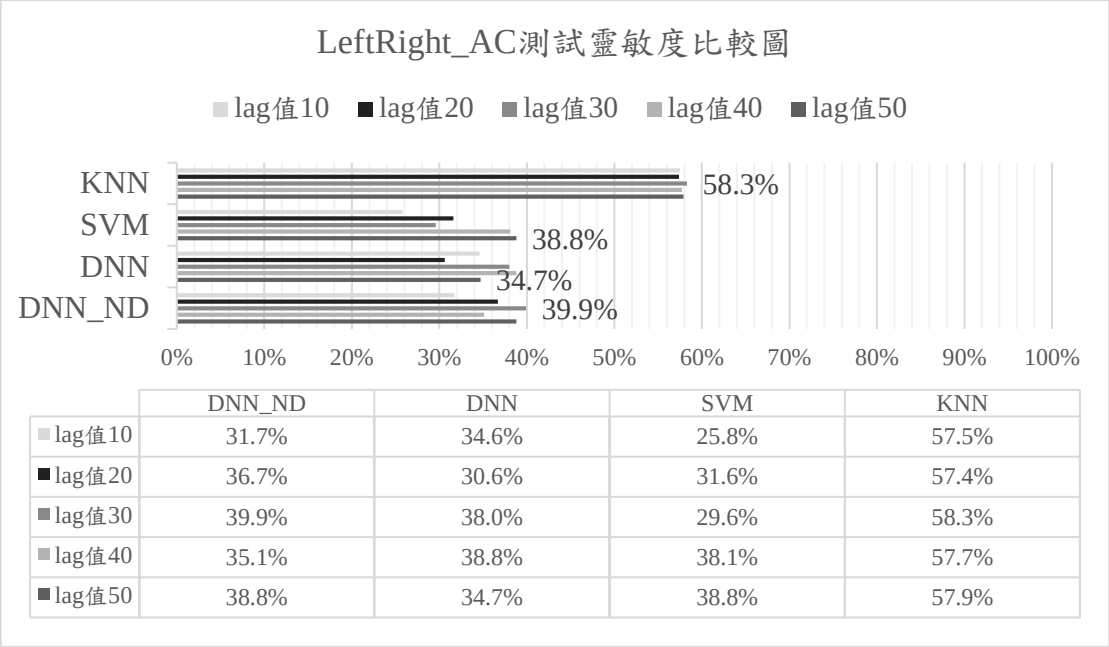
	DNN_ND	DNN	SVM	KNN
lag值10	65.2%	63.7%	67.3%	65.8%
lag值20	65.2%	64.8%	68.3%	65.3%
lag值30	65.0%	64.8%	67.9%	66.0%
lag值40	66.0%	65.3%	67.7%	66.1%
lag值50	65.0%	66.0%	67.3%	65.9%

(圖 16) : LeftRight_AC 測試準確率比較圖



	DNN_ND	DNN	SVM	KNN
lag值10	46.7%	44.3%	51.9%	60.0%
lag值20	47.2%	45.8%	54.2%	59.1%
lag值30	47.1%	46.5%	53.3%	60.1%
lag值40	48.7%	47.5%	52.1%	60.0%
lag值50	46.9%	48.6%	51.2%	59.8%

(圖 17) : LeftRight_AC 測試精確率比較圖



	DNN_ND	DNN	SVM	KNN
lag值10	31.7%	34.6%	25.8%	57.5%
lag值20	36.7%	30.6%	31.6%	57.4%
lag值30	39.9%	38.0%	29.6%	58.3%
lag值40	35.1%	38.8%	38.1%	57.7%
lag值50	38.8%	34.7%	38.8%	57.9%

(圖 18) : LeftRight_AC 測試靈敏度比較圖

由(表 21)(表 22)可以看到我們用準確率判斷時，DNN 不管是在 Ushuffle 方法，又或是在 LeftRight 方法的情況下都相對高了一點，但綜觀來看，也可以說 SVM、DNN、DNN_ND 的結果其實是差不多的，而 KNN 效果最差。

(表 21)：各分類器在 Ushuffle 方法下測試準確率最高

Classfier	**偽造蛋白質不交互作用發法**	**蛋白質描述方法**	**測試準確率**
KNN	Ushuffle	RAAA1-lt2	85.0%
SVM	Ushuffle	AC(lag=10)	94.5%
DNN	Ushuffle	RAAA2-lt3	95.5%
DNN_ND	Ushuffle	RAAA2-lt3	**95.6%**

(表 22)：各分類器在 LeftRight 方法下測試準確率最高

Classfier	**偽造蛋白質不交互作用方法**	**蛋白質描述方法**	**測試準確率**
KNN	LeftRight	RAAA2-lt2	66.3%
SVM	LeftRight	RAAA1-lt2	71.1%
DNN	LeftRight	RAAA1-lt2	**72.9%**

DNN_ND	LeftRight	RAAA2-lt3	71.4%

由(表 23)(表 24)可以看到我們用精確率判斷時，雖然在使用 LeftRight 偽造蛋白質不交互作用對下，KNN 的精確率是最高的，但其實和 SVM、DNN、DNN_ND 的結果差不多，而在使用 Ushuffle 偽造蛋白質不交互作用對下，KNN 精確率是最低的，其他也差異不大，整體看下來 KNN 一樣是最差的。

(表 23)：各分類器在 Ushuffle 方法下測試精確率最高

Classfier	**偽造蛋白質不交互作用發法**	**蛋白質描述方法**	**測試準確率**
KNN	Ushuffle	RAAA1-lt2	85.1%
SVM	Ushuffle	AC(lag=50)	94.4%
DNN	Ushuffle	RAAA1-lt2	**96.3%**
DNN_ND	Ushuffle	RAAA2-lt3	95.5%

(表 24)：各分類器在 LeftRight 方法下測試精確率最高

Classfier	**偽造蛋白質不交互作用方法**	**蛋白質描述方法**	**測試準確率**
KNN	LeftRight	RAAA2-lt2	**61.4%**
SVM	LeftRight	RAAA1-lt2	58.7%
DNN	LeftRight	RAAA1-lt2	58.9%
DNN_ND	LeftRight	RAAA2-lt3	60.9%

由(表 25)(表 26)可以看到我們用靈敏度判斷時，雖然在使用 LeftRight 偽造蛋白質不交互作用對下，KNN 和 DNN 的靈敏度是比較高的，但如果再細看其他數據會發現，KNN 還是比較好的，而在使用 Ushuffle 偽造蛋白質不交互作用對下，KNN 靈敏度是最低的，其他也差異不大，整體看下來 KNN 一樣不是很好。

(表 25)：各分類器在 Ushuffle 方法下測試靈敏度最高

Classfier	**偽造蛋白質不交互作用發法**	**蛋白質描述方法**	**測試準確率**
KNN	Ushuffle	RAAA1-lt2	85.5%
SVM	Ushuffle	AC(lag=50)	94.6%

DNN	Ushuffle	RAAA3-lt3	95.4%
DNN_ND	Ushuffle	RAAA1-lt2、RAAA2-lt3	**95.6%**

(表 26)：各分類器在 LeftRight 方法下測試靈敏度最高

Classfier	**偽造蛋白質不交互作用方法**	**蛋白質描述方法**	**測試準確率**
KNN	LeftRight	RAAA2-lt2	60.8%
SVM	LeftRight	RAAA1-lt2	44.9%
DNN	LeftRight	RAAA1-lt2	**61.4%**
DNN_ND	LeftRight	RAAA1-lt2	53.8%

第六章 結論

在本篇論文中，我們對 STRING 蛋白質交互作用資料庫中記載的綠膿桿菌資料進行分類預測，在偽造蛋白質不交互作用對的方法上使用了 Ushuffle 和 LeftRight 兩種，在蛋白質描述方法上我們用了簡化胺基酸(RAAA)和 5 種 lag 值的協方差，最後選用 KNN、SVM、DNN 三種分類器來做最後機器學習測試，同時將 DNN 模型分為是否有 DropOut 層(DNN_ND)的區別來進行比對。

根據實驗結果顯示出在蛋白質描述方法上，簡化胺基酸(RAAA)相比協方差(AC)有高一點的準確率、精確率和靈敏度，但差異不會太大。在偽造蛋白質不交互作用對的方法上，Ushuffle 相比 LeftRight 有較高的準確率、精確率和靈敏度。分類器上，在 DNN 模型中有加入 DropOut 層(DNN)，雖然一般來說可以稍微減緩過擬和的發生，但就實驗數據看下來只有稍微好一點的效果，論文所使用的 DNN 模型還有可以改進的地方。

在使用 RAAA 作為蛋白質描述方法時，DNN 的系統測試準確性效果相對比較好，但使用協方差(AC)作為蛋白質描述方法時，SVM 的系統測試準確性效果會比較好，而 KNN 準確性不管在哪種蛋白質描述方法幾乎都略輸於 SVM 和 DNN。從精確率和靈敏度的角度來看，在 Ushuffle 方法下，一樣是 DNN 和 SVM 優於 KNN，KNN 大部分效果都很差，但在 LeftRight 方法下，KNN 的效果會是最好的。看下來 DNN 和 SVM 實驗結果的表現是比較好的，KNN 還有再調整進步的空間。

最終實驗結果，最高準確率是在使用 Ushuffle 偽造非交互作用蛋白質對方法，蛋白質描述方法 RAAA2-lt3 及機器學習模型 DNN_ND 的 95.6%。最高精確率是在使用 Ushuffle 偽造非交互作用蛋白質對方法，蛋白質描述方法 RAAA1-lt2 及機器學習模型 DNN 的 96.3%。最高靈敏度是在使用 Ushuffle 偽造非交互作用蛋白質對方法，蛋白質描述方法 RAAA1-lt2、RAAA2-lt3 及機器學習模型 DNN_ND 的 95.6%。整體比較準確率、精確率和靈敏度，在本次實驗結果中，會覺得 DNN 效果是最好的。未來可以加入更多種類的偽造蛋白質不交互作用方法以及蛋白質描述方法來進行比較。

參考文獻

[1] E. Banin, D. Hughes, and O. P. Kuipers, "Editorial: Bacterial pathogens, antibiotics and antibiotic resistance," FEMS Microbiology Reviews, vol. 41, no. 3, pp. 450–452, May 2017.

[2] H. Nikaido, "Multidrug Resistance in Bacteria," Annual Review of Biochemistry, vol. 78, no. 1, pp. 119–146, Jun. 2009.

[3] H. F. Lodish and National Library Of Medicine, Molecular cell biology, 4th ed. New York: W.H. Freeman ; Basingstoke, 2000.

[4] S. Jones and J. M. Thornton, "Principles of protein-protein interactions., " Proceedings of the National Academy of Sciences, vol. 93, no. 1, pp. 13–20, Jan. 1996.

[5] P. Melius and J. Yon-Ping Sheng, "Thermal condensation of a mixture of six amino acids," Bioorganic Chemistry, vol. 4, no. 4, pp. 385–391, Dec. 1975.

[6] J. L. Hansen, T. M. Schmeing, P. B. Moore, and T. A. Steitz, "Structural insights into peptide bond formation," Proceedings of the National Academy of Sciences, vol. 99, no. 18, pp. 11670–11675, Sep. 2002.

[7] J. Tanaka, N. Doi, H. Takashima, and H. Yanagawa, "Comparative characterization of random-sequence proteins consisting of 5, 12, and 20 kinds of amino acids, " Protein science, vol. 19, no. 4, pp. 786–795, Mar. 2010.

[8] R. F. Boyer, "Concepts in biochemistry," Hoboken, Nj: Wiley, 2006.

[9] P. Echenique, "Introduction to protein folding for physicists, " Contemporary Physics, vol. 48, no. 2, pp. 81–108, Mar. 2007.

[10] Y. Hou and G. Wu, "Nutritionally Essential Amino Acids," Advances in Nutrition, vol. 9, no. 6, pp. 849–851, Sep. 2018.

[11] G. Wu et al., "Dietary requirements of 'nutritionally non-essential amino acids' by animals and humans," Amino Acids, vol. 44, no. 4, pp. 1107–1113, Dec. 2012.

[12] H. Nielsen, S. Brunak, and G. von Heijne, "Machine learning approaches for the prediction of signal peptides and other protein sorting signals," Protein Engineering, Design and Selection, vol. 12, no. 1, pp. 3–9, Jan. 1999.

[13] K.-C. Chou, "Some remarks on protein attribute prediction and pseudo amino acid composition, " Journal of Theoretical Biology, vol. 273, no. 1, pp. 236–247, Mar. 2011.

[14] Y. Guo, L. Yu, Z. Wen, and M. Li, "Using support vector machine combined with auto covariance to predict protein–protein interactions from protein sequences, " Nucleic Acids Research, vol. 36, no. 9, pp. 3025–3030, Apr. 2008.

[15] S. Fields and O. Song, "A novel genetic system to detect protein–protein interactions, " Nature, vol. 340, no. 6230, pp. 245–246, Jul. 1989.

[16] M. Hirst, C. Ho, L. Sabourin, M. Rudnicki, L. Penn, and I. Sadowski, “A two-hybrid system for transactivator bait proteins,” Proceedings of the National Academy of Sciences, vol. 98, no. 15, pp. 8726–8731, Jul. 2001.

[17] G. Rigaut, A. Shevchenko, B. Rutz, M. Wilm, M. Mann, and B. Séraphin, "A generic protein purification method for protein complex characterization and proteome exploration, " Nature Biotechnology, vol. 17, no. 10, pp. 1030–1032, Oct. 1999.

[18] M. Guttman, G. N. Betts, H. Barnes, M. Ghassemian, P. van der Geer, and E. A. Komives, "Interactions of the NPXY microdomains of the low density lipoprotein receptor-related protein 1," PROTEOMICS, vol. 9, no. 22, pp. 5016–5028, Nov. 2009.

[19] J.-S. Lin and E.-M. Lai, "Protein–Protein Interactions: Co-Immunoprecipitation," Methods in Molecular Biology, vol. 1615, pp. 211–219, 2017.

[20] H. Zhu, "Global Analysis of Protein Activities Using Proteome Chips," Science, vol. 293, no. 5537, pp. 2101–2105, Jul. 2001.

[21] J. Ptacek et al., "Global analysis of protein phosphorylation in yeast," Nature, vol. 438, no. 7068, pp. 679–684, Dec. 2005.

[22] M. Pellegrini, E. M. Marcotte, M. J. Thompson, D. Eisenberg, and T. O. Yeates, "Assigning protein functions by comparative genome analysis: Protein phylogenetic profiles," Proceedings of the National Academy of Sciences, vol. 96, no. 8, pp. 4285–4288, Apr. 1999.

[23] R. Overbeek, M. Fonstein, M. D’Souza, G. D. Pusch, and N. Maltsev, "Use of Contiguity on the Chromosome to Predict Functional Coupling," In Silico Biology, vol. 1, no. 2, pp. 93–108, Jan. 1999.

[24] E. M. Marcotte, "Detecting Protein Function and Protein-Protein Interactions from Genome Sequences," Science, vol. 285, no. 5428, pp. 751–753, Jul. 1999.

[25] A. J. Enright, I. Iliopoulos, N. C. Kyrpides, and C. A. Ouzounis, "Protein interaction maps for complete genomes based on gene fusion events," Nature, vol. 402, no. 6757, pp. 86–90, Nov. 1999.

[26] P. Aloy and R. B. Russell, "Interrogating protein interaction networks through structural biology," Proceedings of the National Academy of Sciences, vol. 99, no. 9, pp. 5896–5901, Apr. 2002.

[27] P. Aloy and R. B. Russell, "InterPreTS: protein Interaction Prediction through Tertiary Structure," Bioinformatics, vol. 19, no. 1, pp. 161–162, Jan. 2003.

[28] Utkan Ogmen, Ozlem Keskin, A Selim Aytuna, R. Nussinov, and Attila Gursoy, "PRISM: protein interactions by structural matching," Nucleic acids research, vol. 33, no. Web Server, pp. W331–W336, Jul. 2005.

[29] T.-W. . Huang et al., "POINT: a database for the prediction of protein-protein interactions based on the orthologous interactome," Bioinformatics, vol. 20, no. 17, pp. 3273–3276, Jun. 2004.

[30] J. Espadaler, O. Romero-Isart, R. M. Jackson, and B. Oliva, "Prediction of protein-protein interactions using distant conservation of sequence patterns and structure relationships," Bioinformatics, vol. 21, no. 16, pp. 3360–3368, Jun. 2005.

[31] L. Nanni and A. Lumini, "An ensemble of K-local hyperplanes for predicting protein-protein interactions," Bioinformatics, vol. 22, no. 10, pp. 1207–1210, Feb. 2006.

[32] Y. Guo, L. Yu, Z. Wen, and M. Li, "Using support vector machine combined with auto covariance to predict protein–protein interactions from protein sequences," Nucleic Acids Research, vol. 36, no. 9, pp. 3025–3030, Apr. 2008.

[33] L. Salwinski, "The Database of Interacting Proteins: 2004 update," Nucleic Acids Research, vol. 32, no. 90001, pp. D449-D451, Jan. 2004.

[34] C. Stark, "BioGRID: a general repository for interaction datasets," Nucleic Acids Research, vol. 34, no. 90001, pp. D535–D539, Jan. 2006.

[35] D. Szklarczyk et al., "STRING v11: protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets," Nucleic Acids Research, vol. 47, no. Database issue, pp. D607–D613, Jan. 2019.

[36] A. Kamburov, K. Pentchev, H. Galicka, C. Wierling, H. Lehrach, and R. Herwig, "ConsensusPathDB: toward a more complete picture of cell biology," Nucleic Acids Research, vol. 39, no. suppl_1, pp. D712–D717, Nov. 2010.

[37] M. Jiang, J. Anderson, J. Gillespie, and M. Mayne, "uShuffle: A useful tool for

shuffling biological sequences while preserving the k-let counts," BMC Bioinformatics, vol. 9, no. 1, Apr. 2008.

[38] J. E. Hill, "cpnDB: A Chaperonin Sequence Database," Genome Research, vol. 14, no. 8, pp. 1669–1675, Aug. 2004.

[39] W. Li and A. Godzik, "Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences," Bioinformatics, vol. 22, no. 13, pp. 1658–1659, May 2006.

[40] K.-C. Chou, "Prediction of protein cellular attributes using pseudo-amino acid composition," Proteins: Structure, Function, and Genetics, vol. 43, no. 3, pp. 246–255, 2001.

[41] A. D. Solis and S. Rackovsky, "Optimized representations and maximal information in proteins," Proteins, vol. 38, pp. 49–164, 2000.

[42] C. Etchebest, C. Benros, Aurélie Bornot, Anne-Claude Camproux, and Alexandre, "A reduced amino acid alphabet for understanding and designing protein adaptation to mutation," European Biophysics Journal, vol. 36, no. 8, pp. 1059–1069, Jun. 2007.

[43] M.Ababdi, P. Barham, J.Chen, Z.Chen, A.Davis. J.Dean.etc. "TensorFlow: A System for Large-Scale Machine Learning," pp. 265-283, 2016.

[44] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," arXiv:1207.0580 [cs], Jul. 2012.

[45] W. S. Noble, "What is a support vector machine?," Nature Biotechnology, vol. 24, no. 12, pp. 1565–1567, Dec. 2006.

[46] D. M. Hawkins, "The Problem of Overfitting," Journal of Chemical Information and Computer Sciences, vol. 44, no. 1, pp. 1–12, Jan. 2004.

[47] L.-Y. Hu, M.-W. Huang, S.-W. Ke, and C.-F. Tsai, "The distance function effect on k-nearest neighbor classification for medical datasets," SpringerPlus, vol. 5, no. 1, Aug. 2016.

[48] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," arXiv.org, Dec. 22, 2014.