Aphanchanok Ousuwan

# Refinement of a graph convolutional neural network approach applied to classify cancer types

Final Master's Project

Directed by: Dr. Carme Julià Ferré

UNIVERSITAT ROVIRA i VIRGILI

Tarragona
2022

## Abstract

The graph convolution neural network(GCNN) model was developed in order to predict the type of cancers or the normal tissue which depending on the profile of gene expression. The Cancer Genome Atlas (TCGA) dataset has provided the data of 10,340 tumor samples from 33 cancer types and 731 normal samples from various tissues of origin.

The objective of the master's thesis is to improve the results of the network architecture used in "Classification of Cancer Types Using Graph Convolutional Neural Networks" by Ricardo Ramirez et al. and "Graph Convolution Neural Networks applied to classify cancer types" by Heribert Saldana. Specifically, the main objective is to refine the classification of cancer classes that present problems. One more objective is to verify the post-modeling analysis to obtain more accurate results.

In this paper, the network architecture of four GCNN models based on protein-to-protein interaction (PPI) graph, protein-to-protein interaction plus singleton (PPIS) graph, co-expression (COEX) graph and co-expression plus singleton (COEXS) graph have been refined by changing parameters and network architecture. The prediction accuracy of four GCNN models achieved the outstanding of prediction accuracies (89-96%) among 34 classes. The problems of the prediction error in some types of cancer were high. The results obtained in this work show that the prediction accuracy of GCNN model of PPIS, COEX and COEXS are 95-96% and the predictive errors in original model are reduced considerably. Furthermore, the post-modeling analysis algorithm has been validated.

# Table of Contents

# Introduction

In regarding to World Health Organization (WHO), the main cause of death worldwide is Cancer, counting from nearly 10 million deaths in 2020 [1]. The most frequent in 2020 (in the course of new cases of cancer) were breast cancer (2.26 million cases), lung cancer (2.21 million cases), colorectal cancer (1.93 million cases), prostate cancer (1.41 million cases), skin (non-melanoma) (1.20 million cases) and stomach cancer (1.09 million cases). The most frequent causes of cancer death in 2020 were; lung cancer (1.80 million cases), colorectal cancer (935,000 cases), liver cancer (830,000 cases), stomach cancer (769,000 cases) and breast cancer (685,000 cases) [1].

According to the American Association for cancer research (AACR) shown the cost of national cancer-attributable in the United States are projected to increase by over 30 percent from 2015 to 2030. The costs were estimated by cancer site, the stage of cancer and the stage of treatment which included the primary stage determined as the first 12 months referring to the cancer determination; the extinction stage which determined as 12 months before the extinction and the proceeding stage determined as the month in the middle of the primary stage and the extinction stage consist of the observation duration surrounded by those who extinct from any other causes[2]. As a result, the primary stage of screening and classifying the cancer types before rising symptoms have essential effected to the social and economic issue.

## Background

There was a report regarding the process of computing for classifying the cancer type and the stage of cancer. The classification of cancer types using graph convolution neural networks [3] and graph convolutional neural networks applied to classify cancer types [4] are developed a graph convolution neural network (GCNN) model which classified the typical tissue and 33 types of cancer from the data file which randomly commanding the gene by the cancer genome atlas (TCGA). GCNN was developed the model data specified in non-Euclidean domains such as graph [5]. GCNN discharge convolution on the data graph throughout the graph Laplacian in preference of on the fixed grid of 1-D or 2-D Euclidean organized data. GCNN models have been adapted to predict metastatic breast cancer issue and to combine the protein-protein interaction database (STRING) into breast cancer study [6-9]. GCNN models have been influenced me to research the classification of expression-based cancer type.

There had been recommended and skilled four GCNN models in order to examine the cancer type predict and classify the cancer-specific markers; by using the integrated number of TCGA gene definition data sets, along with 10,340 tumor samples from 33 cancer types and 731 normal samples from numerous tissue of origin. The four model graphs were created, namely, the protein-to-protein interaction (PPI) network, the protein-to-protein interaction plus singleton (PPIS), the co-expression (COEX) network and the co-expression plus singleton (COEXS) network. The models had been projected successfully classified tumor samples without distraction from normal

tissue samples which recommending the markers probably cancer specific without counting on the tissues. In addition, they investigated the co-expression graph model and caused of each gene on the precision of cancer type prediction using in silico gene disorder in which they set up the one gene's definition level 0 or 1 in one sample before fed into the set up models per imitate and then investigate the disruption in precision of cancer type prediction. They expected that the largest changes in the precision of cancer type prediction could be differential the marker genes to indicate the cancer type.

### Objectives of the master's thesis

The objective of the master's thesis is to improve the results of the network architecture used in "Classification of Cancer Types Using Graph Convolutional Neural Networks" by Ricardo Ramirez et al. and "Graph Convolution Neural Networks applied to classify cancer types" by Heribert Saldana. Specifically, the main objective is to refine the classification of cancer classes that present problems which are colon adenocarcinoma (COAD) with rectum adenocarcinoma (READ) and stomach adenocarcinoma (STAD) with esophageal carcinoma (ESCA). One more objective is to verify the post-modeling analysis to obtain more accurate results.

## Proposed method

### 2.1 Data Preparation

In the article [3] Ricardo Ramirez et al. uses the data from TCGA. The RNA-seq data were downloaded from TCGA and been clarified as reported previously [10]. The database consists of the completed collection of 11,071 samples which composed of 10,340 samples from 33 of cancer types and 731 of normal samples from 23 different types of tissues in which of those 18 samples had no consist of origin tissue that specified as non-cancer in December 2018. All abbreviations consist of distinct the number of cancers and the normal samples in which shown in the Table 1 in particular; normal tissue samples in the READ study stand for normal rectum tissue. The primary measure unit is fragments per kilo base million (FPKM) which the originators assign the logarithm to the base two and adding one to the value which higher than 0.6. In addition, all values are standardized in the middle of 0 and 1 which relevant to the method shown detail on the web [12] or in the paper [13]. Two different input graphs were created such as a co-expression graph and PPI graph from the STRING database (https://string-db.org/) [14, 15]. The data is created in square matrix which stands for a finite graph (adjacency matrix) by using MATLAB. The components of matrix had shown the adjoining pairs of vertices in the graph.

### 2.1.1 PPI graph (PPI)

All 7,091 genes were fed into the BioMart data base in order to discover the similar unusual Ensembl protein IDs [17]. As a consequence of the survival of non-coding genes in TCGA data set and the restricted number of proteins in the STRING data base in which 4,444 genes were elected to construct the graph. The certain reliance of the genes connection in the STRING data base was considered. Therefore, any of connections among two genes was considered, a weight of 1 will be settled in an adjacency matrix. The PPI graph is stand for a 4,444 by 4,444 adjacency matrix. The string data base is elected for the PPI collaboration caused the amount and potential of data coverage, comfortably visualization support and user-friendly file exchange format [18].

### 2.1.2 PPI graph created by singleton nodes (PPIS)

All 7,091 genes were used in PPI and singleton node graph which all 2,647 genes exclude in the PPI graph was treated as singleton nodes as shown in Figure 1. The 7,091 by 7,091 adjacency matrix consist of the 4,444 by 4,444 adjacency matrix from the PPI graph at the upper-left corner and zeros in the other positions.

### 2.1.3 Co-expression graph (COEX)

All 7,091 genes were using for generating the co-expression graph, Spearman correlation was estimated using MATLAB to create a correlation matrix in the middle of each gene in the data set. Spearman Correlation is extensively adopted the standard to evaluate the monotonic linear or non-linear relationships in sequencing data [16]. If the correlation among two genes is greater than 0.6 with a p is less than 0.05 in which 1 is placed in an adjacency matrix or if not 0 is placed in an adjacency matrix. If there is no correlation is greater than 0.6 with a given gene in addition the gene will be displaced from the gene list, leading to the total of 3,866 genes in the co-expression graph. The graph layout is stand for a 3,866 by 3,866 adjacency matrix.

### 2.1.4 Co-expression graph created by singleton nodes (COEXS)

The co-expression and singleton graph and the supplementary of 3,225 genes which exclude in the co-expression graphs are included as singleton nodes where co-expression upper-left corner and zeros in the other places to create a 7,091 by 7,091 adjacency matrix. Eventually, all input data in the prior study are PPI, PPIS, COEX and COEXS.

| Tumor Type | Cohort | Numbers of cancer samples | Numbers of normal samples |
|---|---|---|---|
| Adrenocortical carcinoma | ACC | 79 | 0 |
| Bladder urothelial carcinoma | BLCA | 414 | 19 |
| Breast invasive carcinoma | BRCA | 1108 | 113 |
| Cervical and endocervical cancers | CESC | 306 | 3 |
| Cholangiocarcinoma | CHOL | 36 | 9 |
| Colon adenocarcinoma | COAD | 478 | 41 |
| Lymphoid Neoplasm Diffuse Large B-cell Lymphoma | DLBC | 48 | 0 |
| Esophageal carcinoma | ESCA | 162 | 11 |
| Glioblastoma multiforme | GBM | 168 | 0 |
| Head and Neck squamous cell carcinoma | HNSC | 502 | 44 |
| Kidney Chromophobe | KICH | 65 | 24 |
| Kidney renal clear cell carcinoma | KIRC | 539 | 72 |
| Kidney renal papillary cell carcinoma | KIRP | 289 | 32 |
| Acute Myeloid Leukemia | LAML | 151 | 0 |
| Brain Lower Grade Glioma | LGG | 528 | 0 |
| Liver hepatocellular carcinoma | LIHC | 374 | 50 |
| Lung adenocarcinoma | LUAD | 535 | 59 |
| Lung squamous cell carcinoma | LUSC | 502 | 49 |
| Mesothelioma | MESO | 86 | 0 |
| Ovarian serous cystadenocarcinoma | OV | 379 | 0 |
| Pancreatic adenocarcinoma | PAAD | 178 | 4 |
| Pheochromocytoma and Paraganglioma | PCPG | 183 | 52 |
| Prostate adenocarcinoma | PRAD | 499 | 3 |
| Rectum adenocarcinoma | READ | 166 | 10 |
| Sarcoma | SARC | 263 | 2 |
| Skin Cutaneous Melanoma | SKCM | 471 | 1 |
| Stomach adenocarcinoma | STAD | 374 | 32 |
| Testicular Germ Cell Tumors | TGCT | 142 | 0 |
| Thyroid carcinoma | THCA | 508 | 58 |
| Thymoma | THYM | 119 | 2 |
| Uterine Corpus Endometrial Carcinoma | UCEC | 552 | 23 |
| Uterine Carcinosarcoma | UCS | 56 | 0 |
| Uveal Melanoma | UVM | 80 | 0 |

***Table 1*** *All 11,071 samples which composed of 10,340 samples from 33 of cancer types and 731 of normal samples from 23 different types of tissues.*

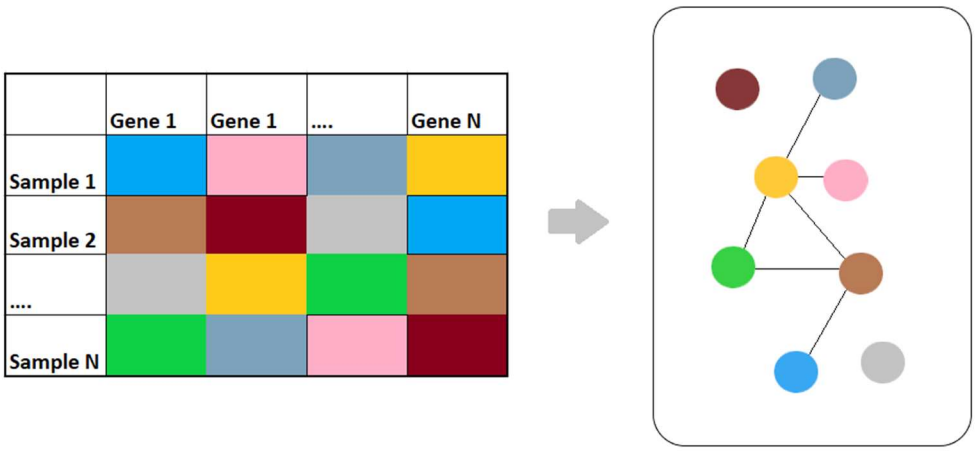


***Figure 1*** *The RNA-seq data from TCGA and generated in graph. Obviously, the connections among two genes and singleton node.*

## 2.2 Graph theory

### 2.2.1 ChebNet

A graph is stand for $G = (V, E)$ where $V$ is the nodes (vertices) a finite set of $|V| = n$ vertices, $E$ is edges severally and $A$ *is* $A_{ij} \in \mathbb{R}^{n \times n}$. $A$ a weighted adjacency matrix encoding the connection weight in the middle of two vertices. The input graphs of this program are indirect. The adjacency matrix A showed how nodes are connected:

$$A_{ij} := \begin{cases} 1, & if\ there\ is\ an\ edge\ connecting\ from\ node\ i\ to\ node\ j \\ 0, & otherwise \end{cases}$$

$A$ is a symmetric matrix for an undirected graph. The degree matrix $D$ is a diagonal matrix where $D \in \mathbb{R}^{n \times n}$ , with the elements of $D_{ii}$ denotes the number of neighbours for node $i$ in undirected matrix. The operation performing on this node is called the filter. The graph Laplacian, or Kirchhoffmatrix, is identified by

$$L = D - A \tag{1}$$

The elements of $L$ are given by

$$L_{ij} := \begin{cases} deg\ (V_i), & if\ i = j \\ -1, & if\ i \neq j\ and\ V_i\ is\ adjacent\ to\ V_j \\ 0, & otherwise \end{cases}$$

The normalized graph Laplacian is

$$L = I_n - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}\ where\ I_n\ is\ the\ identity\ matrix \tag{2}$$

The graph Laplacian is the most significant matrix in graph convolutional neural network. It is analogous to the Laplacian operator in Euclidean space. The Laplacian is certainly diagonalized by the Fourier basis $U = [u_0, \dots, u_{n-1}] \in \mathbb{R}^{n \times n}$

Such as $L = U\Lambda U^T$ where $\Lambda = diag([\lambda_0, \dots, \lambda_{n-1}]) \in \mathbb{R}^{n\times n}$ is a diagonal matrix constitute of the eigenvalues of $L$ [19]. Such decomposition admits a spectral-domain operation similar to the Fourier transform in the Euclidean. In this report $x \in \mathbb{R}^{n\times n}$, where $x$ is mapped to a graph $G$. The application of a filter $G$ to the input signal $x$ on the graph can be computed by the convolution of $G$ and $x$, which can be computed in the spectral domain according to in the following equation,

$$y = g(L)x = g(U\Lambda U^T)x = U_g(\Lambda)U_x^T \tag{3}$$

$g_\theta$ is the spectral representation of the filter that gets increasingly complex with the dimension of the input data and the number of neighboring nodes. A non-parametric filter, i.e. a filter whose parameters are all free, would be defined as

$$g_\theta(\Lambda) = diag(\theta) \tag{4}$$

where the parameter $\theta \in \mathbb{R}^n$ is a vector of Fourier coefficients. Polynomial parametrization for localized filters. There are however two limitations with nonparametric filters: 1 they are not localized in space and 2 their learning complexity is in $O_{(n)}$, the dimensionality of the data. These issues can be overwhelmed with the use of a polynomial filter in order to decrease the complication, a polynomial expansion of $g$ can be obtained as

$$g_\theta(\Lambda) = \sum_{K=0}^{K-1} \theta_k \Lambda_k \tag{5}$$

where $\Lambda = diag([\lambda_0^k, \dots, \lambda_{n-1}^k])$ and $\theta_k$ are the polynomial coefficients. It is shown [19] that this expansion yields local filters with manageable calculation. The Chebyshev approximation $T_m(x)$ of order $m$ have been proposed in Defferrard et al. for this expansion and is shown by

$$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x) \; with \; T_0 = 1 \; and \; T_1 = x \tag{6}$$

where $T_0(x) = 1$ and $T_1(x) = x$ [19, 20]. The spectral filter is now given by a truncated Chebyshev polynomial:

$$g_\theta(\Lambda) = \sum_{K=0}^{K-1} \theta_k \mathrm{T}_k(\Lambda') \tag{5}$$

where, $\theta \in \mathbb{R}^k$ now stand for a vector of the Chebyshev coefficients, the $\Lambda'$ denotes the rescaled $\Lambda$. $\lambda_{max}$ denotes the largest eigenvalue of $L$. Scaling is done as

$$\Lambda' = \frac{2\Lambda}{\lambda_{max} - I_n} \tag{6}$$

that maps the eigenvalues in [−1,1]. This makes the Chebyshev expansion to have $x'_0 = x$ and $x'_1 = L'_x$ which exceedingly decreased the computational cost. The consequence of this appliance is known as ChebNet.

### 2.2.2 Graph Convolutional Network

The GCNN model includes an input graph represented by the adjacency matrix, graph convolutional layer (coarsening and pooling), and a hidden layer fully connected to a Softmax output layer. The formular (2) can be reduced into the new form base on " Semi-supervised classification with graph convolutional networks"[21]

$$y = \theta \left( I_n + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \right) x \tag{7}$$

$$L' = L + 1 \; and \; D'_{ii} = \sum_j A'_{ij} \tag{8}$$

So, the final expression for filtering is

$$y = \theta \left( D'^{-\frac{1}{2}} A' D'^{-\frac{1}{2}} \right) x \tag{9}$$

Finally, this resulting implementation is also referred to as graph convolutional network (GCN).

The convolution layers in Convolutional Neural Networks, ‘convolution’ in GCNs is basically the same operation. It refers to multiplying the input neurons with a set of weights that are commonly known as filters. The filters act as a sliding window across the whole image and enable CNNs to learn features from neighboring cells. Within the same layer, the same filter will be used throughout image as shown in Figure 2.

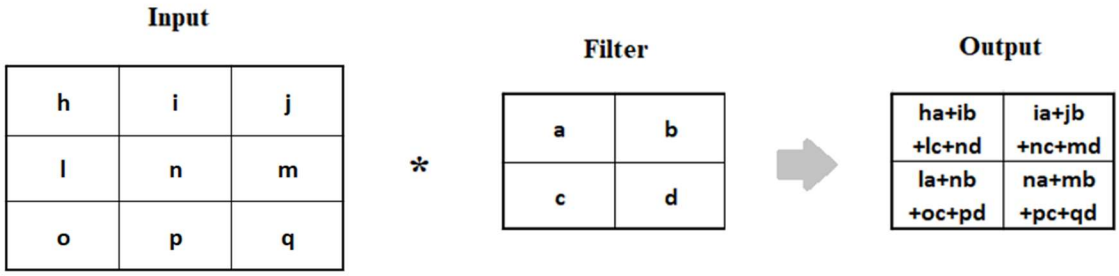


***Figure2*** *Example of convolution operator.*

## 2.3 Model architecture

In original article [3] the graph convolutional neural network (GCNN) was developed recently to model data defined in non-Euclidean domains such as graphs [3-4]. Input is one dimensional gene expression levels of TCGA samples and the input graph is adjacency matrix of genes. The graph is then pooled into a single GCNN layer to be fed into the hidden and output layers. A greedy algorithm was used for layer coarsening, which reduced the number of nodes roughly by half. A greedy algorithm is any algorithm that follows the problem-solving heuristic of making the locally optimal choice at each stage. In many problems, a greedy strategy does not produce an optimal solution, but a greedy heuristic can yield locally optimal solutions that approximate a globally optimal solution in a reasonable amount of time [24]. The greedy rule chose an unselected node to be paired with another unpaired neighbor node and their vertices being summed together. The pooling and coarsening a singleton node, the node grouped with a random node that was unpaired. The output nodes of the final GCNN layer served as the input to a single dense fully connected layer with a ReLu function which then led to the output layer with a Softmax function to get the probabilities as shown in Figure 3[3].

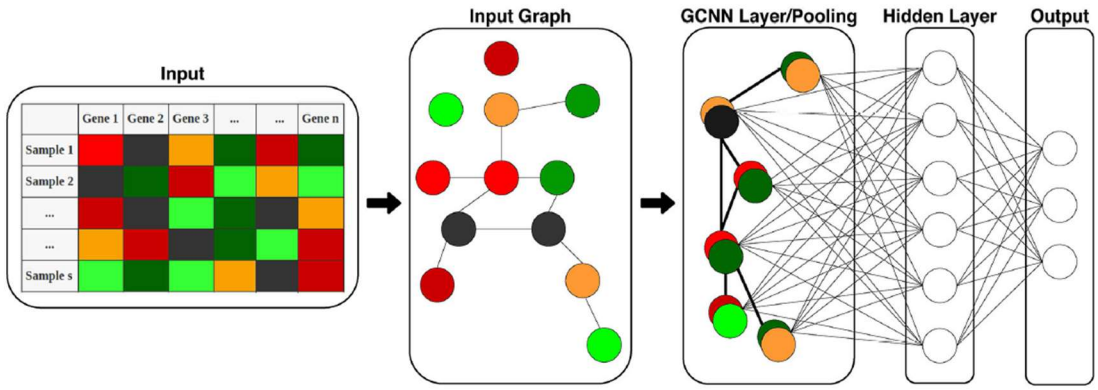

***Figure 3*** *Structure of the proposed GCNN model. The model includes two parts: graph convolution and a fully connected output layer for classification. Input is 1D gene expression levels of TCGA samples and the adjacency matrix of genes (input graph). The graph is then pooled into a single GCNN layer to be fed into the hidden and output layers [3].*

## 2.4 Problems of original model

Regarding to the model architecture in 2.3, the previous author concluded one point that requires further research to improve the prediction accuracies because the best model is PPIS GCNN has got 94% accuracy. This number is low in the medical environment [4]. The results obtained in the previous work show the prediction errors are high between COAD, READ and STAD, ESCA. Regarding to the confusion matrix in the paper [4], the number of prediction error between COAD and READ is the highest as shown in Figure 8. It may be caused by their proximity the COAD is colon adenocarcinoma and the READ is rectum adenocarcinoma [4].

Refinement of model architecture

The utilization of convolution neural network is working in layers which heap collectively as deep as designed. The GCNN including an input graph stand for the adjacency matrix, graph convolution layer (coarsening and pooling), and a hidden layer which fully combined to a Softmax output layer as shown in Figure 4.

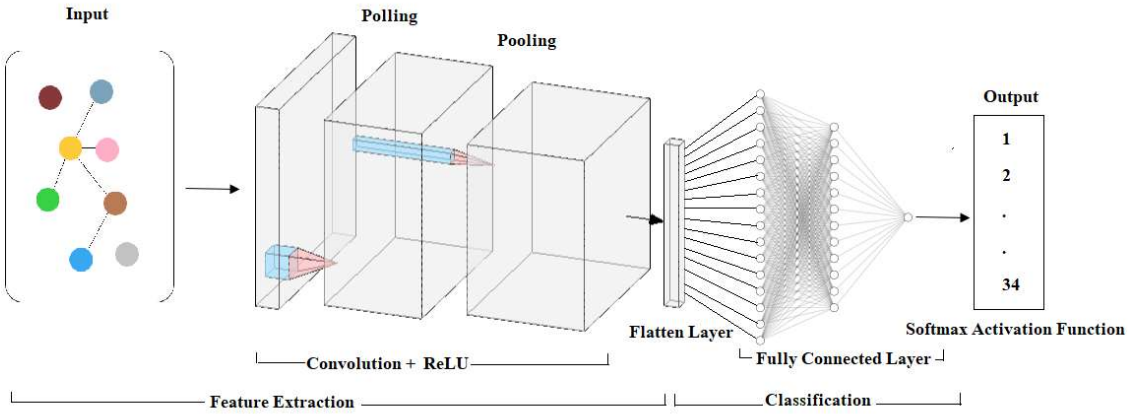


***Figure 4*** *GCNN input graph stand for the adjacency matrix, then graph convolution layer (coarsening and pooling), and a hidden layer which fully combined to a Softmax output layer.*

In order to improve the problem-solving in the between of COAD, READ and STAD, ESCA in the original architecture, the model architecture has been refined. On the other hand, the types of

cancer that already good should be good as always. The 32 experiments were set in addition perform 10 times per experiment. The data were progressed in order to fit the pooled graph, the data and the labels are divided into 80% for the training of GCNN and 20% of remaining for validation and test. The prediction of parameter corresponding for creating GCNN depended on the analysis; the composition of GCNN will be created to receive the list with the graph and the parameters as in the input. Once, GCNN is created and trained using the 80 percent of data over 20 epochs, the validation data (remaining 20%) in order to contradict the result of training up and adjust the learning rate depended on the analysis rhythmically. The training instantly done, GCNN is ready to classify data which two convolution layers (coarsening and pooling) will be used as shown in Figure 5 and the greedy algorithm will be decreased roughly the nodes number by half. The regulation of greedy selects the deselected node to be combined with another unpaired neighbor node and their apex being summed together. The pooling and coarsening singleton node with a random group node was unpaired. The output node of the final GCNN layer served as the single input which fully connected with ReLu function which controlled the possibility output layer of Softmax function.

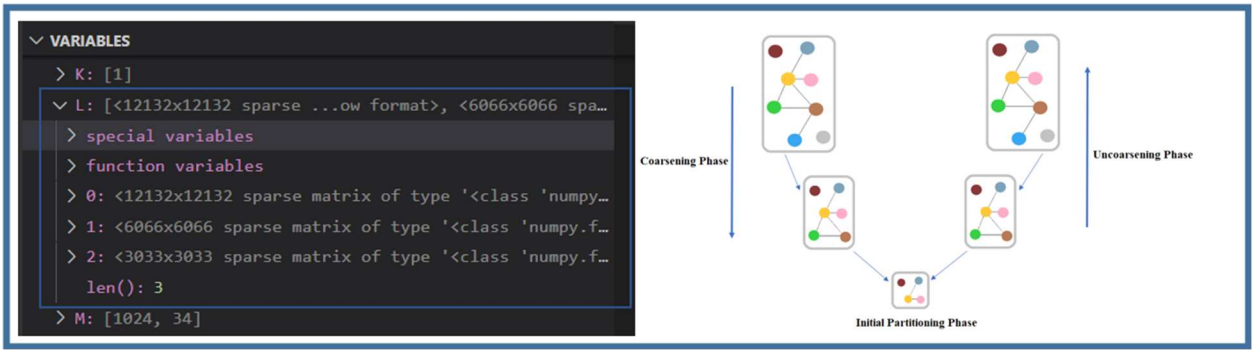


***Figure 5*** *Convolution layer.*

## Post-modelling Analysis of GCNN Model

In original article [3] the post-modelling analysis aims at examination how much the predictions of a trained model changed before and after a gene was perturbed in computer simulations, where significant prediction accuracy change suggested the importance of the gene in the classification. To calculate the contribution score of each gene to 34 classification types. The post-modelling algorithm will set each gene to the lowest value (0) and do the production. Then again will set the same gene to the highest value (1) and do the production as shown in Figure 6. To see how the expression change would affect the prediction accuracy of the trained model for each sample [4]. The newly obtained prediction accuracies caused by a gene will be compared to the original prediction accuracy from the model for the cancer type labelled by TCGA data. The larger prediction accuracy change of the labelled cancer type was chosen as a contribution score of that gene for that cancer type. The process was repeated for each gene in all cancer types and normal samples, resulting in a contribution score for each gene of all 34 classification groups (33 for

cancer types and normal type). The contribution scores will be represented by a matrix with dimensions of the number of classes (34) by the number of genes. Then, the final contributions will be normalized to their respective class resulting in their gene affected the score between zero to one.

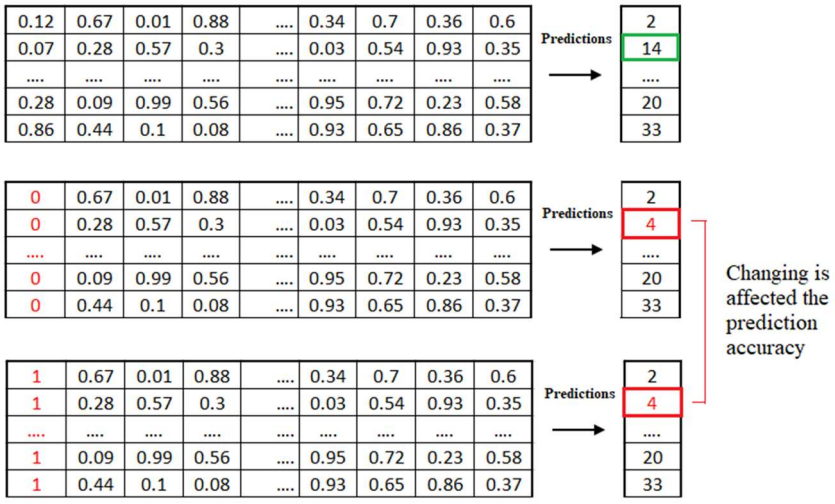


***Figure 6*** *The post-modelling algorithm will set each gene to 0 and do the production. Then again will set the same gene to 1 and do the production.*

## Problem of post-modeling analysis

In the original code [3]. there was a hard code of 7100 in post-modeling analysis is shown in Figure 7 which is fixed by the pervious author[4]. The number of genes in the protein-to-protein graphs post modelling was not correct. The protein to protein has 7091 genes in the code the loop reaches the gene 7099. After modified and computed the code process 7447 genes. This process takes a few hours to complete. This step is needed to verify in order to obtain more accurate results.

```
305:### starting the dropout code
306: Knockdown = []
307: Knockup = []
308: for i in range(7100):
309:     if i % 50 == 0:
310:         print(' On gene number')
311:         print(i)
312:     Datadown = test_data
313:     Datadown[:,i]=0
314:     Knockdown.append(model.evaluate(Datadown, test_labels, pre = True))
315:     Dataup = test_data
316:     Dataup[:,i]=0
317:     Knockup.append(model.evaluate(Dataup, test_labels, pre = True))
318: sio.savemat('Knockdown.mat', {'Knockdown':Knockdown})
319: sio.savemat('Knockup.mat', {'Knockup':Knockup})
```

***Figure 7*** *A part of post-modelling in the original code [4].*

# Results

## Previous results

According to the results of paper [4] which shown the calculation of means and standard deviation for the accuracy in ten processing. The results derived from "Graph Convolutional Neural Networks applied to classify cancer types" are represented in the 1st results[3]. The 2nd and 3rd are the results from the previous author[4]. The prediction accuracy of GCNN model is shown in Table 2. As the results of four models in the first, second and the third of PPI plus singleton GCNN model accomplish the best on average. There is the constant of lowest standard deviation (93.82% ± 1.17%, mean ± std) and (93.96% ± 0.55%, mean ± std). The result of first and second of prediction accuracy for the PPI GCNN model is the lowest (81.94% ± 8.66%, mean ± std) and (84.67% ± 5.37%, mean ± std).

| | The 1st results from the classification of cancer types using graph convolution neural networks | The 2nd results from graph convolutional neural networks applied to classify cancer types | The 3rd results using random data 70% train, 15% validation and 15% testing |
|---|---|---|---|
| Protein to protein interaction | 81.94 ± 8.66 | 84.67 ± 5.37 | 87.76 ± 1.85 |
| Protein to protein interaction plus singleton | 93.82 ± 1.17 | 93.96 ± 0.55 | 93.81 ± 0.66 |
| Co-expression | 92.57 ± 2.73 | 91.72 ± 2.68 | 80.25 ± 2.68 |
| Co-expression plus singleton | 93.32 ± 0.41 | 93.05 ± 0.78 | 93.07 ± 0.37 |

***Table 2*** *The validation accuracy performed 10 times in graph convolutional neural networks applied to classify cancer types [4].*

Figure 8 shows the confusion matrix from the original model of protein-to-protein interaction plus singleton [4]. As the results in Table 2, the PPIS GCNN model is performed the best prediction accuracy on average, but the prediction error is high in some type of cancer. This confusion result matrix shown the problem in between of COAD, READ and STAD, ESCA which READ is rectum adenocarcinoma, COAD is colon adenocarcinoma, STAD is stomach adenocarcinoma and ESCA is esophageal carcinoma. The colon and rectum are parts of the large intestine as shown in Figure 9. In order to be achieved in the research objectives, the consideration of the result of COAD, READ and STAD, ESCA in confusion matrix had been found out that the prediction is one part of high-risk making mistake.

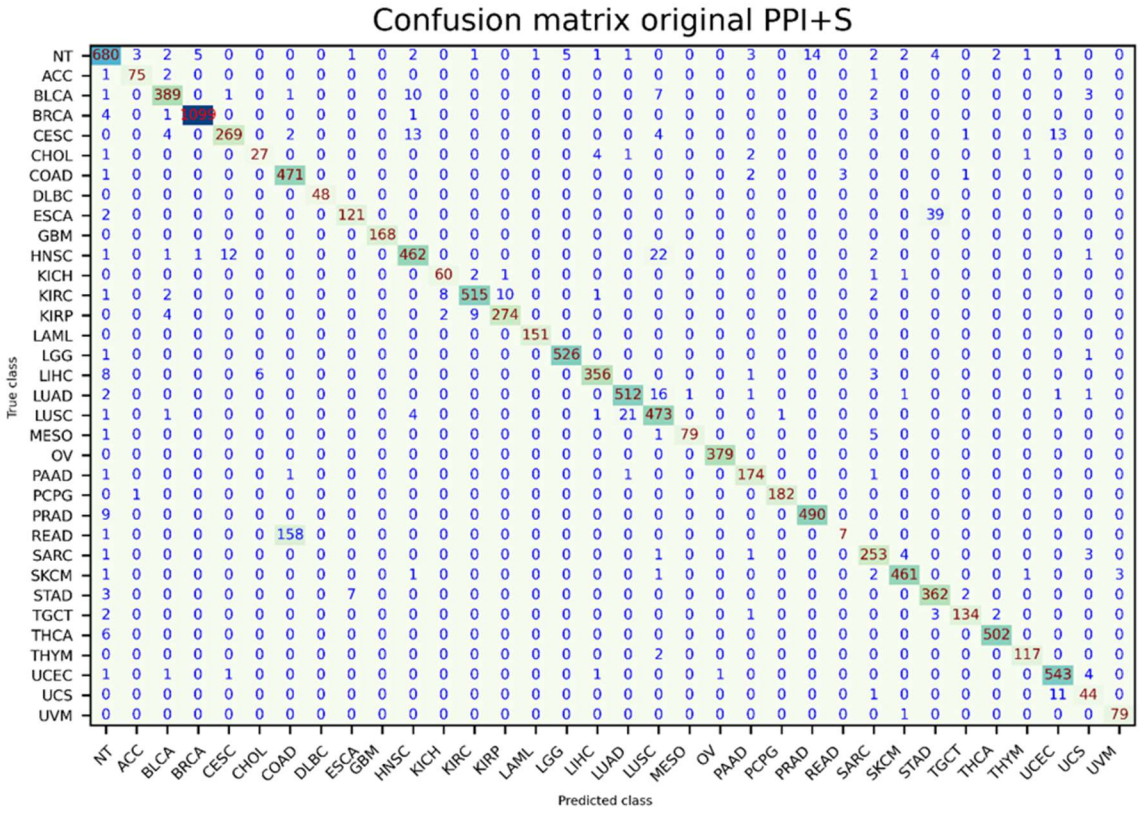


***Figure 8*** *The PPIS GCNN model in the original network model is the best prediction accuracy among 4 models. The highest prediction error is the pair of COAD and READ [3-4].*

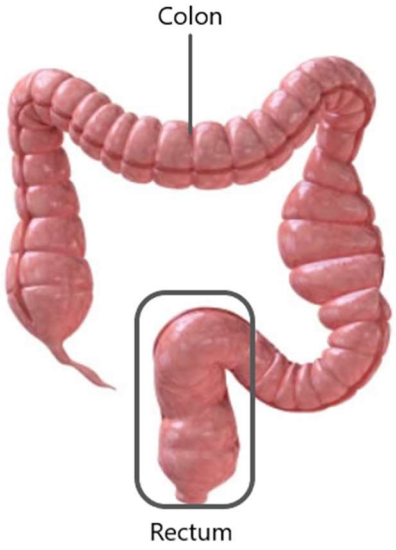


***Figure 9*** *The colon and rectum are parts of the large intestine. The colon is a U-shaped tube made of muscle, found below the stomach. The rectum is a shorter tube connected to the colon [22].*

## Current results

This work shows the results of all the analysis of neural network model. The input data of nodes number are protein-to-protein interaction is 4444, protein-to-protein interaction plus singleton 7091, co-expression 3866 and co-expression plus singleton 7091. The batch size equal to 200 and the convergence time (epochs) is 20. The coarsening layer performed the pooling using the Metis algorithm [25] which the processing time of the program is 5 to 7 hours for one analysis including the post-modeling analysis of GCNN model. I have performed 10 times per one analysis for calculating mean and the standard deviation. For example: protein-to-protein interaction plus singleton with 2 coarsening layers, 1024 size of hidden layer and learning rate is equal to 0.005 which shown in the Table 3. The mean and the standard deviation for all analysis are represented in Table 4. Regarding to the processing time of the program is 5 to 7 hours based on the model of the laptop. Figure 37 shows the specifications of the laptop that used in the experiments.

| $1^{st}$ | $2^{nd}$ | $3^{rd}$ | $4^{th}$ | $5^{th}$ | $6^{th}$ | $7^{th}$ | $8^{th}$ | $9^{th}$ | $10^{th}$ |
|---|---|---|---|---|---|---|---|---|---|
| 97.66 | 97.27 | 96.81 | 96.68 | 95.09 | 97.4 | 94.45 | 96.28 | 98.36 | 93.48 |

***Table 3*** *The example of protein-to-protein interaction plus singleton with 2 coarsening layers, 1024 size of hidden layer and learning rate is equal to 0.005 performed 10 times per one analysis for calculating mean and the standard deviation.*

Mean equal to 96.348.

Standard Deviation: $s = \sqrt{\frac{\sum_{i=1}^{n}(x_i - x')^2}{n-1}}$

$$s = \sqrt{\frac{SS}{n-1}}$$

$$s = \sqrt{\frac{21.46496}{10-1}}$$

$$s = \sqrt{2.3849956} = 1.54$$

| | Coarsening layer | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 1 | | | | 2 | | | |
| | Size of hidden layer | | | | Size of hidden layer | | | |
| | 512 | | 1024 | | 512 | | 1024 | |
| | Learning rate | | Learning rate | | Learning rate | | Learning rate | |
| | 0.001 | 0.005 | 0.001 | 0.005 | 0.001 | 0.005 | 0.001 | 0.005 |
| Protein-to-protein interaction | 82.024±7.60 | 88.887±1.62 | 88.599±0.59 | 89.248±0.51 | 80.106±6.14 | 81.124±2.39 | 83.603±1.46 | 73.447±3.01 |
| Protein-to-protein interaction plus singleton | 85.801±13.16 | 92.018±2.58 | 76.507±18.29 | 94.958±2.56 | 77.921±14.57 | 94.634±0.97 | 70.436±13.96 | 96.348±1.54 |
| Co-expression | 76.376±18.99 | 95.215±1.35 | 86.796±12.84 | 95.524±1.30 | 70.236±14.97 | 93.901±1.51 | 87.970±12.30 | 94.674±1.35 |
| Co-expression plus singleton | 76.50±16.35 | 95.781±1.41 | 83.249±14.69 | 95.627±2.08 | 84.60±12.98 | 94.546±1.50 | 83.146±13.67 | 95.131±1.44 |

***Table 4*** *Mean and the standard deviation of each GCNN model.*

## Result of protein-to-protein interaction

The original confusion matrix as shown in Figure 10 is represented the original PPI GCNN model, the prediction error of COAD to READ is 156 and STAD to ESCA is 70. The modified confusion matrix as shown in Figure 11 is represented the modified PPI GCNN model by the previous author [4]. The prediction error of COAD to READ is 151 and STAD to ESCA is 64. The confusion matrix of this paper is represented in Figure 12, the prediction error of COAD to READ is 145 and STAD to ESCA is 35 with accuracy 90.36 as shown in Figure 13. The best model of protein-to-protein interaction is the combinations between one coarsening layer, 1024 size of hidden layer and learning rate equal to 0.005.

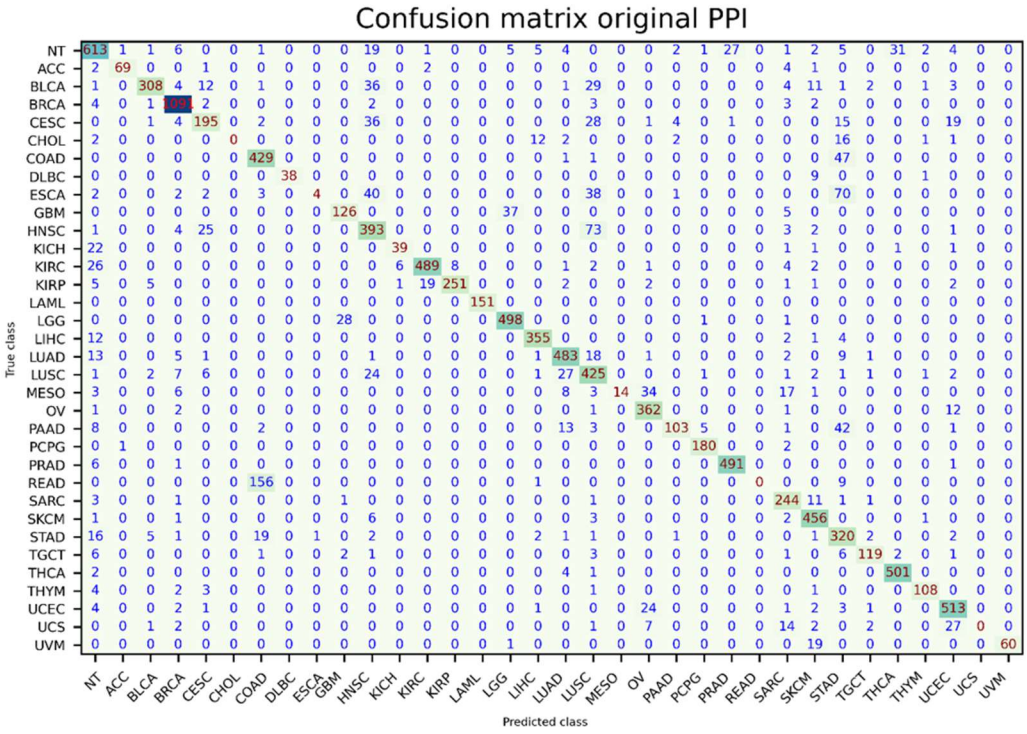


***Figure 10*** *The Original confusion matrix of protein-to-protein interaction in "graph convolutional neural networks applied to classify cancer types"[4].*

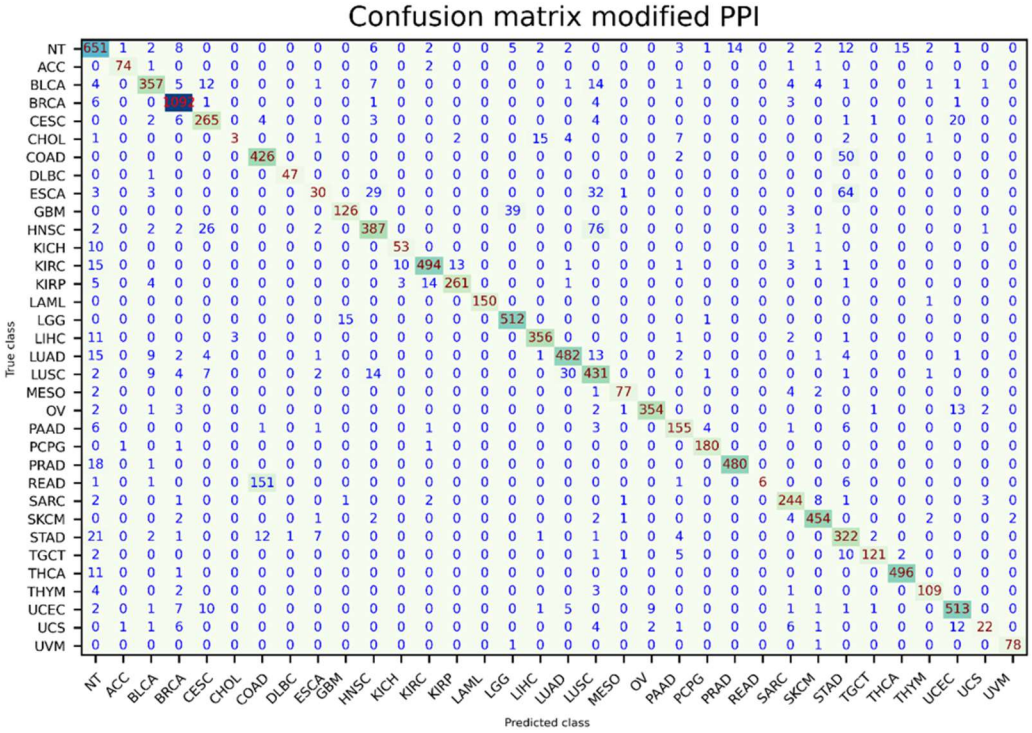


***Figure 11*** *The modified confusion matrix of protein-to-protein interaction in "graph convolutional neural networks applied to classify cancer types"[4].*

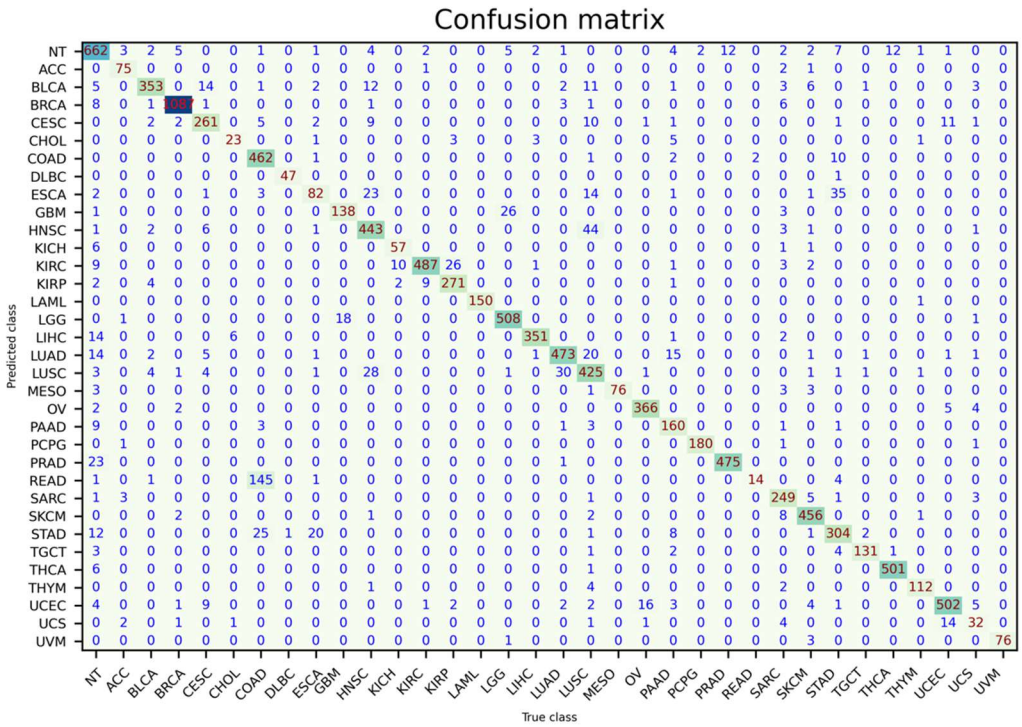


***Figure 12*** *The confusion matrix of refinement of a graph convolutional neural network of protein-to-protein interaction.*

```
step 200 / 889 (epoch 4.50 / 20):
  learning_rate = 2.84e-03, loss_average = 3.04e+00
  validation accuracy: 22.91 (498 / 2174), f1 (weighted): 15.42, loss: 2.96e+00
  time: 105s (wall 23s)
step 400 / 889 (epoch 8.99 / 20):
  learning_rate = 1.62e-03, loss_average = 5.26e-01
  validation accuracy: 81.55 (1773 / 2174), f1 (weighted): 79.99, loss: 7.71e-01
  time: 211s (wall 46s)
step 600 / 889 (epoch 13.49 / 20):
  learning_rate = 9.20e-04, loss_average = 3.94e-01
  validation accuracy: 86.29 (1876 / 2174), f1 (weighted): 85.65, loss: 5.60e-01
  time: 319s (wall 69s)
step 800 / 889 (epoch 17.99 / 20):
  learning_rate = 4.97e-04, loss_average = 3.59e-01
  validation accuracy: 87.63 (1905 / 2174), f1 (weighted): 86.87, loss: 4.66e-01
  time: 426s (wall 91s)
step 889 / 889 (epoch 19.99 / 20):
  learning_rate = 3.85e-04, loss_average = 3.23e-01
  validation accuracy: 89.19 (1939 / 2174), f1 (weighted): 88.58, loss: 4.43e-01
  time: 482s (wall 104s)
validation accuracy: peak = 89.19, mean = 73.51
train accuracy: 90.65 (8064 / 8896), f1 (weighted): 89.91, loss: 3.63e-01
time: 9s (wall 2s)
test  accuracy: 89.19 (1939 / 2174), f1 (weighted): 88.58, loss: 4.43e-01
time: 2s (wall 1s)
Validation  accuracy: 90.36 (10003 / 11070), f1 (weighted): 89.65, loss: 3.79e-01
time: 14s (wall 6s)

(tf) C:\Users\USER\TFM>Python PPI.py PPI
```

***Figure 13*** *The validation accuracy of PPI model one coarsening layer, 1024 size of hidden layer and learning rate equal to 0.005.*

## Result of protein-to-protein interaction plus singleton

The original confusion matrix as shown in Figure 14 is represented the original PPIS GCNN model, the prediction error of COAD to READ is 158 and STAD to ESCA is 39. The modified confusion matrix as shown in Figure 15 is represented the modified PPIS GCNN model by the previous author [4]. The prediction error of COAD to READ is 54 and STAD to ESCA is 15. The confusion matrix of this paper is represented in Figure 16 and the prediction error of COAD to READ is 20 and STAD to ESCA is 12 with accuracy 98.36 as shown in Figure 17. The best model of protein-to-protein interaction plus singleton is the combinations between two coarsening layers, 1024 size of hidden layer and learning rate equal to 0.005.

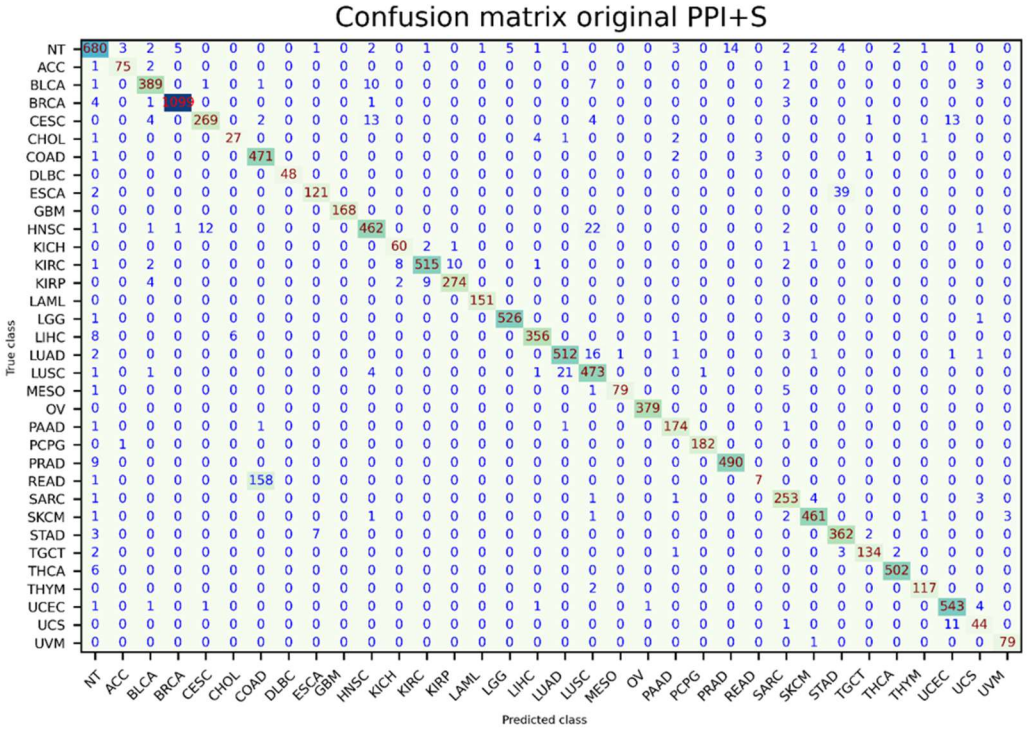


***Figure 14*** *The Original confusion matrix of protein-to-protein interaction plus singleton in "graph convolutional neural networks applied to classify cancer types"[4].*

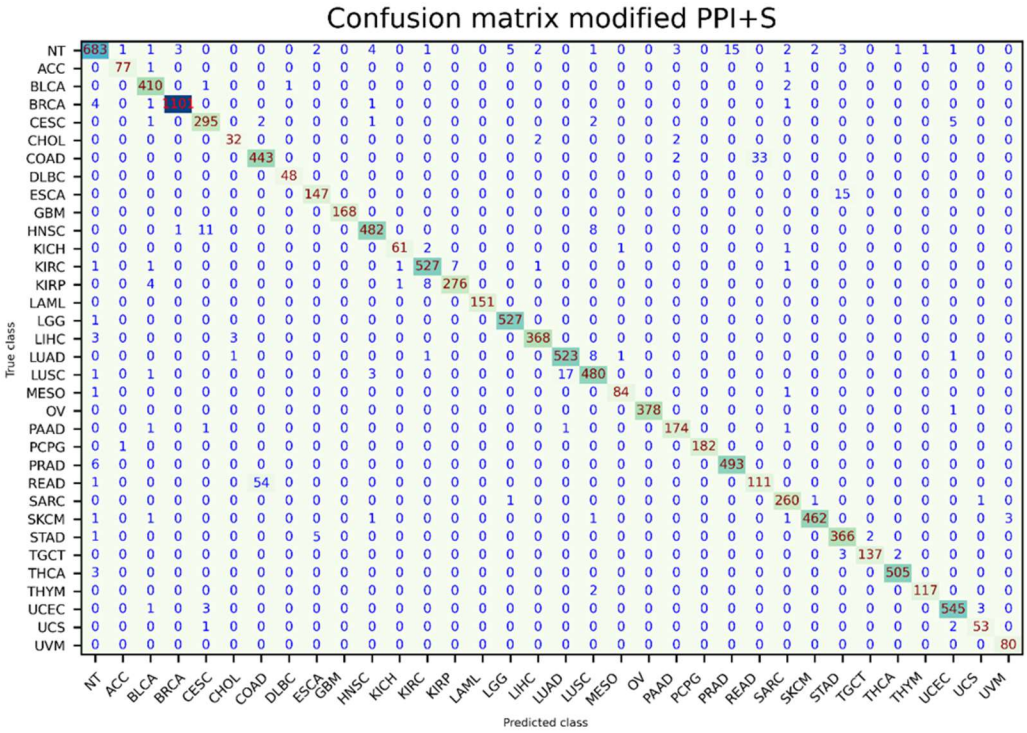


***Figure 15*** *The modified confusion matrix of protein-to-protein interaction plus singleton in "graph convolutional neural networks applied to classify cancer types"[4].*

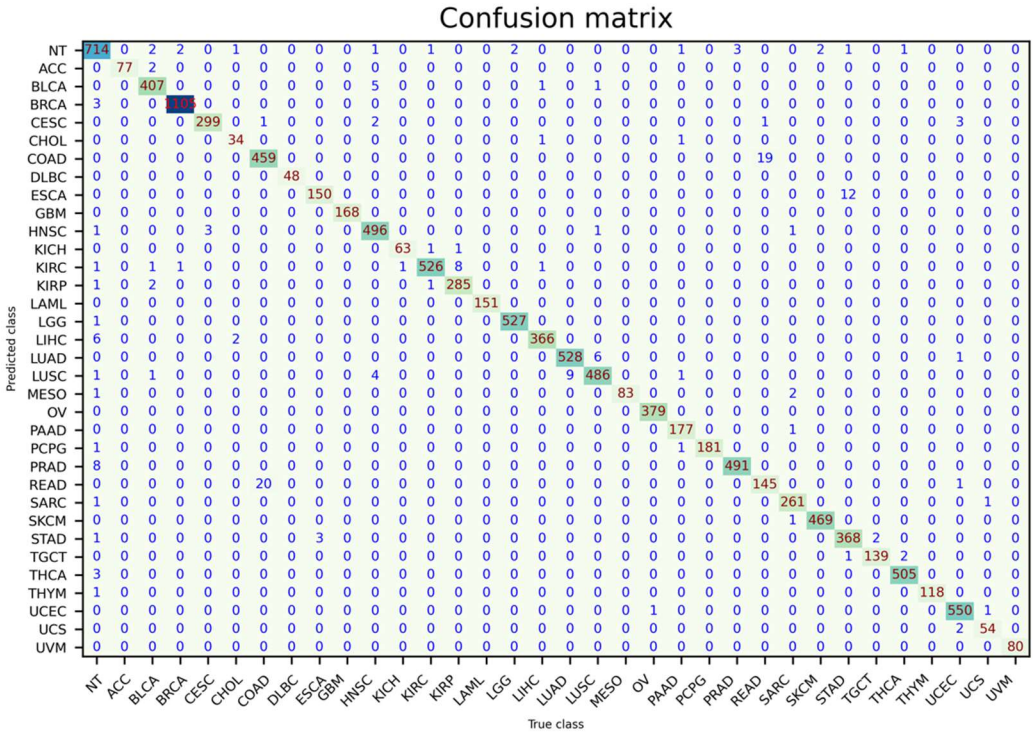


***Figure 16*** *The confusion matrix of refinement of a graph convolutional neural network of protein-to-protein interaction plus singleton.*

```
step 200 / 885 (epoch 4.52 / 20):
  learning_rate = 2.84e-03, loss_average = 1.81e-01
  validation accuracy: 93.42 (2074 / 2220), f1 (weighted): 93.43, loss: 3.84e-01
  time: 166s (wall 32s)
step 400 / 885 (epoch 9.04 / 20):
  learning_rate = 1.62e-03, loss_average = 9.74e-02
  validation accuracy: 94.19 (2091 / 2220), f1 (weighted): 93.82, loss: 3.51e-01
  time: 338s (wall 67s)
step 600 / 885 (epoch 13.56 / 20):
  learning_rate = 9.20e-04, loss_average = 7.22e-02
  validation accuracy: 93.87 (2084 / 2220), f1 (weighted): 93.36, loss: 3.47e-01
  time: 509s (wall 101s)
step 800 / 885 (epoch 18.08 / 20):
  learning_rate = 4.97e-04, loss_average = 4.29e-02
  validation accuracy: 94.82 (2105 / 2220), f1 (weighted): 94.64, loss: 3.35e-01
  time: 683s (wall 136s)
step 885 / 885 (epoch 20.00 / 20):
  learning_rate = 4.05e-04, loss_average = 4.67e-02
  validation accuracy: 94.77 (2104 / 2220), f1 (weighted): 94.71, loss: 3.30e-01
  time: 767s (wall 153s)
validation accuracy: peak = 94.82, mean = 94.22
train accuracy: 99.27 (8785 / 8850), f1 (weighted): 99.27, loss: 6.65e-02
time: 15s (wall 4s)
test  accuracy: 94.77 (2104 / 2220), f1 (weighted): 94.71, loss: 3.30e-01
time: 4s (wall 1s)
Validation  accuracy: 98.36 (10889 / 11070), f1 (weighted): 98.36, loss: 8.97e-02
time: 20s (wall 7s)

(tf) C:\Users\USER\TFM>
```

***Figure 17*** *The validation accuracy of PPI plus singleton model two coarsening layer, 1024 size of hidden layer and learning rate equal to 0.005.*

## Result of co-expression

The original confusion matrix as shown in Figure 18 is represented the original COEX GCNN model, the prediction error of COAD to READ is 122 and STAD to ESCA is 37. The modified confusion matrix as shown in Figure 19 is represented the modified COEX GCNN model by the previous author [4]. The prediction error of COAD to READ is 61 and STAD to ESCA is 17. The confusion matrix of this paper is represented in Figure 20 and the prediction error of COAD to READ is 43 and STAD to ESCA is 19 with accuracy 96.95 as shown in Figure 21. The best model of co-expression is the combinations between one coarsening layer, 1024 size of hidden layer and learning rate equal to 0.005.

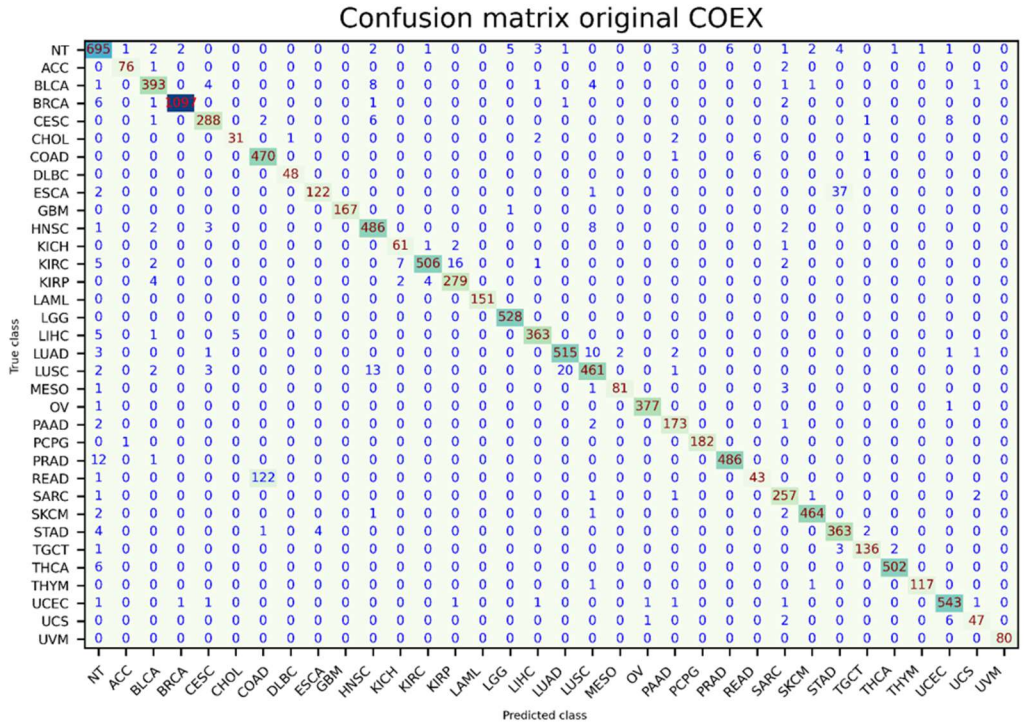


***Figure 18*** *The Original confusion matrix of co-expression in "graph convolutional neural networks applied to classify cancer types"[4].*

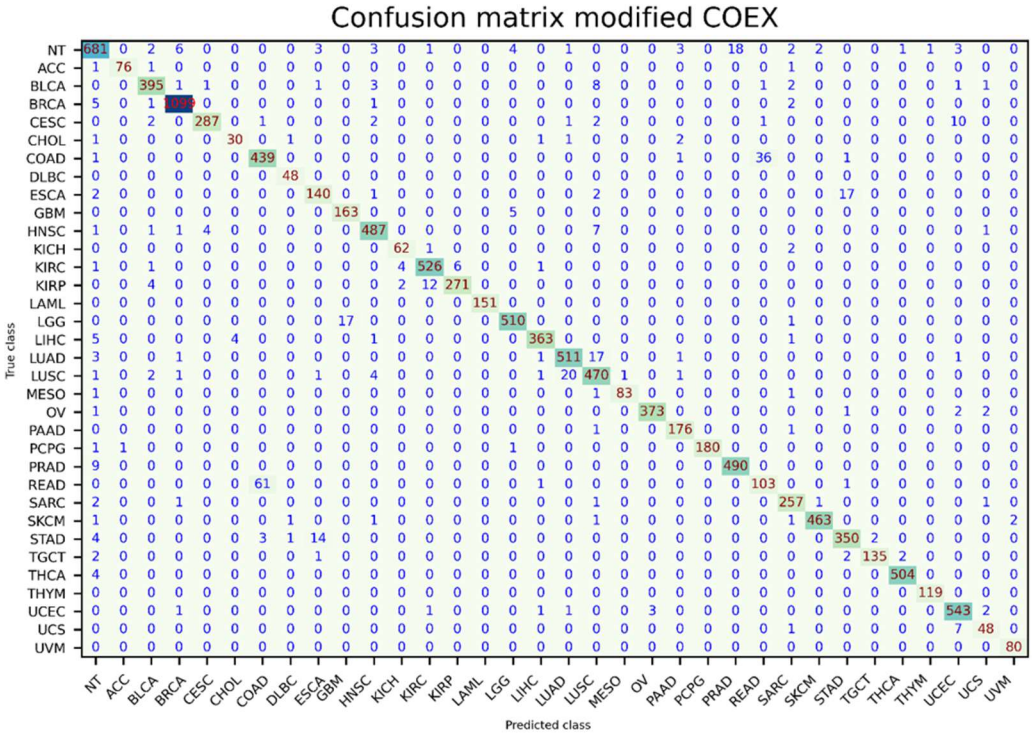


***Figure 19*** *The modified confusion matrix of co-expression in "graph convolutional neural networks applied to classify cancer types"[4].*

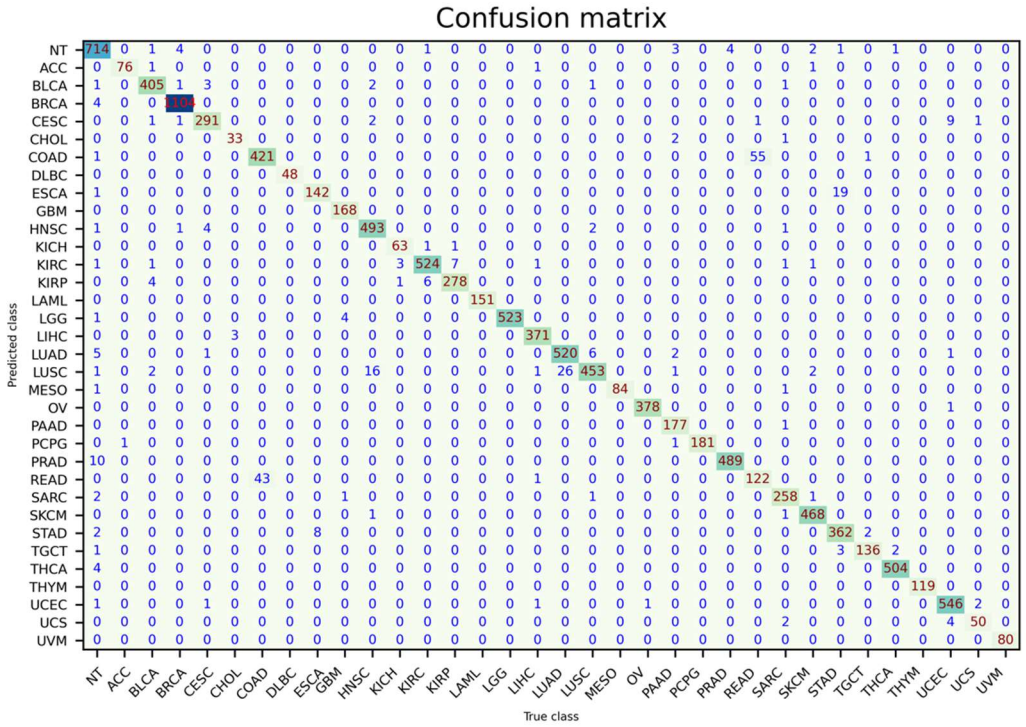


***Figure 20*** *The confusion matrix of refinement of a graph convolutional neural network of co-expression.*

```
step 200 / 885 (epoch 4.52 / 20):
  learning_rate = 2.84e-03, loss_average = 2.03e-01
  validation accuracy: 91.98 (2042 / 2220), f1 (weighted): 91.50, loss: 2.17e+00
  time: 58s (wall 14s)
step 400 / 885 (epoch 9.04 / 20):
  learning_rate = 1.62e-03, loss_average = 1.35e-01
  validation accuracy: 92.97 (2064 / 2220), f1 (weighted): 92.87, loss: 1.84e+00
  time: 118s (wall 28s)
step 600 / 885 (epoch 13.56 / 20):
  learning_rate = 9.20e-04, loss_average = 1.01e-01
  validation accuracy: 94.32 (2094 / 2220), f1 (weighted): 94.29, loss: 1.81e+00
  time: 180s (wall 42s)
step 800 / 885 (epoch 18.08 / 20):
  learning_rate = 4.97e-04, loss_average = 8.41e-02
  validation accuracy: 93.83 (2083 / 2220), f1 (weighted): 93.81, loss: 1.60e+00
  time: 241s (wall 57s)
step 885 / 885 (epoch 20.00 / 20):
  learning_rate = 4.05e-04, loss_average = 8.57e-02
  validation accuracy: 94.28 (2093 / 2220), f1 (weighted): 94.21, loss: 1.65e+00
  time: 271s (wall 64s)
validation accuracy: peak = 94.32, mean = 93.48
train accuracy: 97.62 (8639 / 8850), f1 (weighted): 97.63, loss: 3.84e-01
time: 6s (wall 2s)
test  accuracy: 94.28 (2093 / 2220), f1 (weighted): 94.21, loss: 1.65e+00
time: 2s (wall 1s)
Validation  accuracy: 96.95 (10732 / 11070), f1 (weighted): 96.95, loss: 3.16e-01
time: 10s (wall 5s)

(tf) C:\Users\USER\TFM>
```

***Figure 21*** *The validation accuracy of co-expression model one coarsening layer, 1024 size of hidden layer and learning rate equal to 0.005.*

## Result of co-expression plus singleton

The original confusion matrix as shown in Figure 22 is represented the original COEXS GCNN model, the prediction error of COAD to READ is 100 and STAD to ESCA is 23. The modified confusion matrix as shown in Figure 23 is represented the modified COEXS GCNN model by the previous author [4]. The prediction error of COAD to READ is 139 and STAD to ESCA is 19. The confusion matrix of this paper is represented in Figure 24 and the prediction error of COAD to READ is 53 and STAD to ESCA is 8 with accuracy 96.74 as shown in Figure 25. The best model of co-expression plus singleton is the combinations between one coarsening layer, 1024 size of hidden layer and learning rate equal to 0.005.

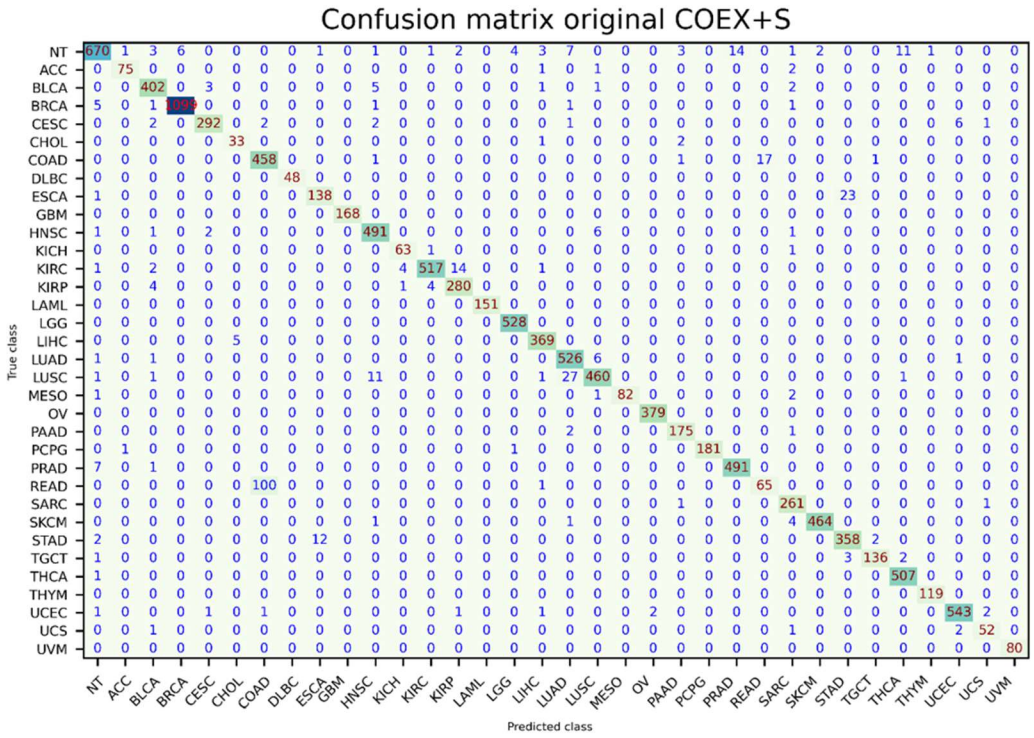


***Figure 22*** *The Original confusion matrix of co-expression plus singleton in "graph convolutional neural networks applied to classify cancer types"[4].*

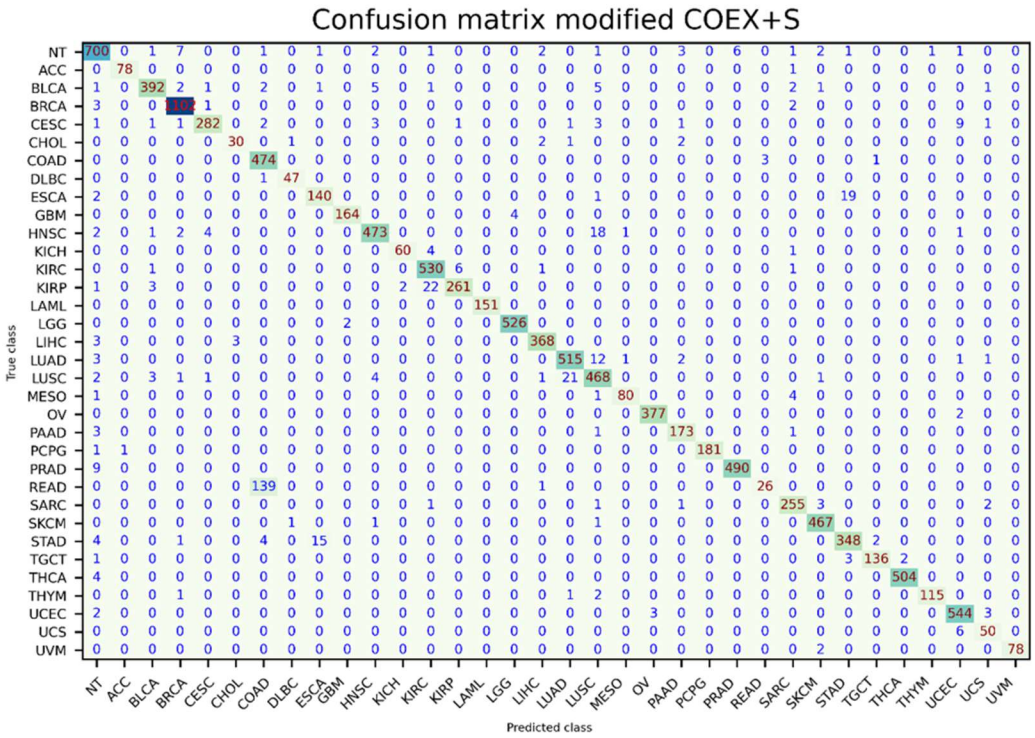


***Figure 23*** *The modified confusion matrix of co-expression plus singleton in "graph convolutional neural networks applied to classify cancer types"[4].*

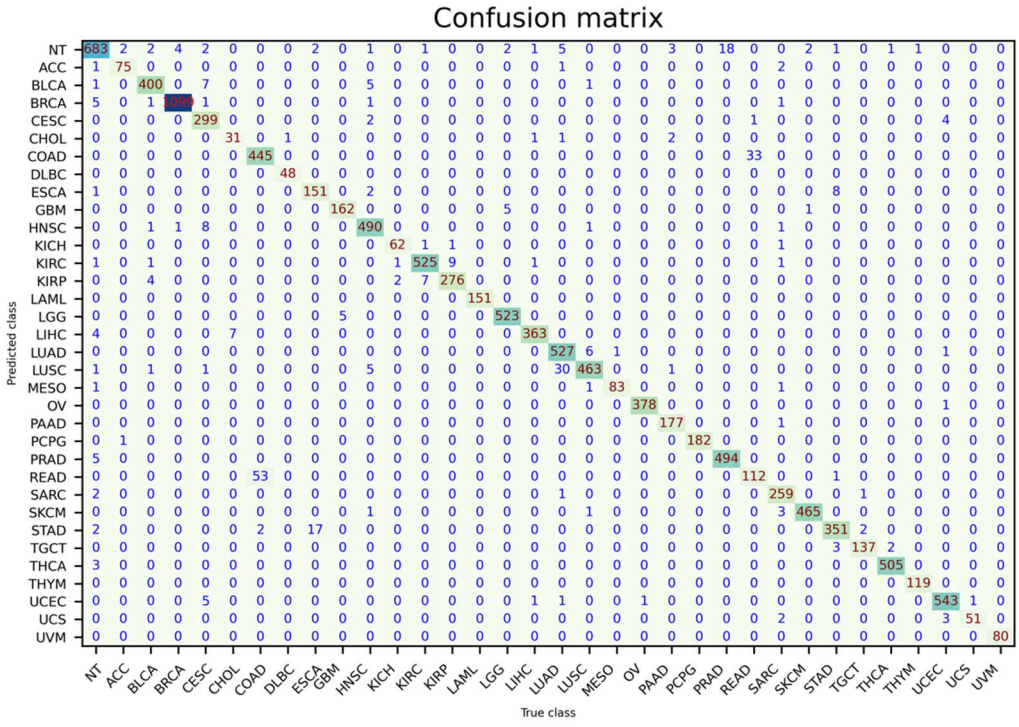


***Figure 24*** *The confusion matrix of refinement of a graph convolutional neural network of co-expression plus singleton.*

```
step 200 / 885 (epoch 4.52 / 20):
  learning_rate = 2.84e-03, loss_average = 2.39e-01
  validation accuracy: 92.07 (2044 / 2220), f1 (weighted): 91.43, loss: 1.71e+00
  time: 62s (wall 14s)
step 400 / 885 (epoch 9.04 / 20):
  learning_rate = 1.62e-03, loss_average = 1.46e-01
  validation accuracy: 93.15 (2068 / 2220), f1 (weighted): 93.18, loss: 1.84e+00
  time: 124s (wall 28s)
step 600 / 885 (epoch 13.56 / 20):
  learning_rate = 9.20e-04, loss_average = 1.09e-01
  validation accuracy: 93.96 (2086 / 2220), f1 (weighted): 93.66, loss: 1.93e+00
  time: 189s (wall 43s)
step 800 / 885 (epoch 18.08 / 20):
  learning_rate = 4.97e-04, loss_average = 8.55e-02
  validation accuracy: 93.47 (2075 / 2220), f1 (weighted): 93.05, loss: 2.06e+00
  time: 255s (wall 58s)
step 885 / 885 (epoch 20.00 / 20):
  learning_rate = 4.05e-04, loss_average = 8.41e-02
  validation accuracy: 93.78 (2082 / 2220), f1 (weighted): 93.65, loss: 2.11e+00
  time: 288s (wall 66s)
validation accuracy: peak = 93.96, mean = 93.29
train accuracy: 97.48 (8627 / 8850), f1 (weighted): 97.47, loss: 4.81e-01
time: 6s (wall 2s)
test  accuracy: 93.78 (2082 / 2220), f1 (weighted): 93.65, loss: 2.11e+00
time: 2s (wall 1s)
Validation  accuracy: 96.74 (10709 / 11070), f1 (weighted): 96.72, loss: 3.84e-01
time: 10s (wall 5s)

(tf) C:\Users\USER\TFM>
```

***Figure 25*** *The validation accuracy of co-expression plus singleton model two coarsening layers, 1024 size of hidden layer and learning rate equal to 0.005.*

## Results of post-modeling analysis

The result of post-modeling is represented in this part which the genes were set to zero and the prediction was performed column by column. Then genes were changed to one and the prediction was performed column by column which the results were compared with the result before changing the value. The process of the post-modeling analysis will create 2 graphs by the first author [3] and improved by the second author [4]. In this work, I have done the verification of code and the result. The hard code 7100 is removed as shown in Figure 26. The size of testing data is 2220 × 12144 as shown in Figure 27. The number 12144 derived from level 0 as shown in Figure 28. At the end of step, the program will generate two files .csv which has the data inside 2220 × 12144.

Those files are the input for generating 2 graphs as shown in Figures 29, 31, 33 and 35 are represented how many time that the result of prediction had been affected by a modification in one genes. In order to calculate the result, the gene of each column data block will be set to zero and the prediction will be performed after that set to one and perform the prediction again. In case of the difference result of both times shown from the original class, the number of Y axis which affected gene will be counted. For the second graph in figures 30, 32, 34 and 36 are represented the position of gene which affected the prediction of genes position.

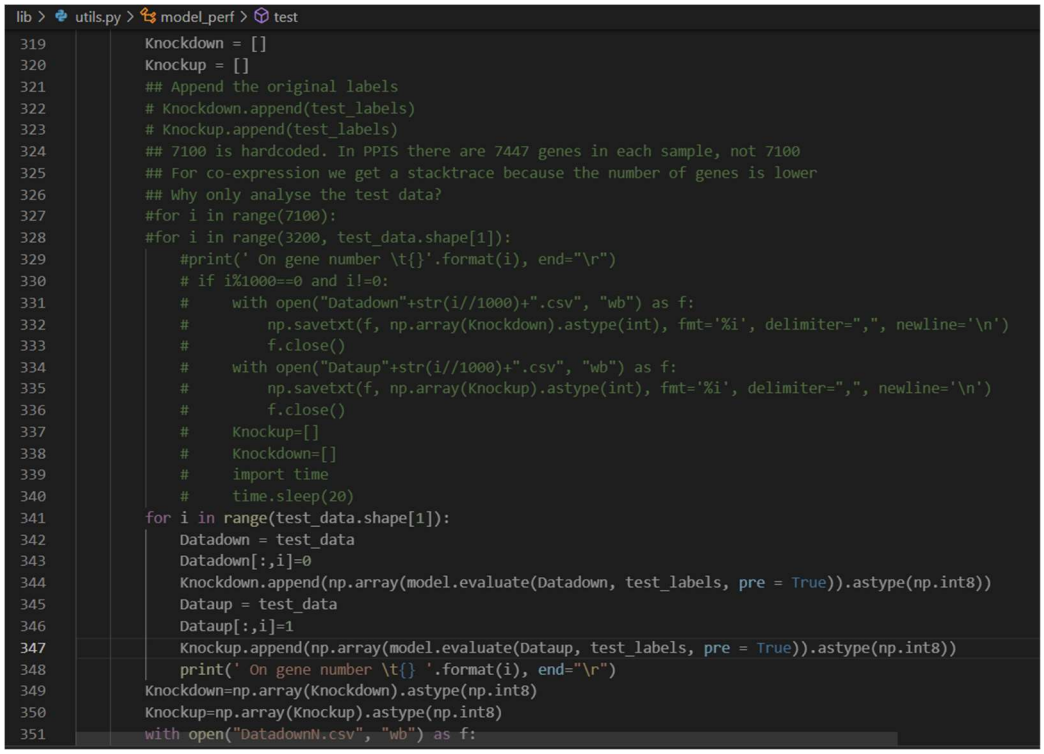


***Figure 26*** *The previous error of post-modelling analysis was fixed.*

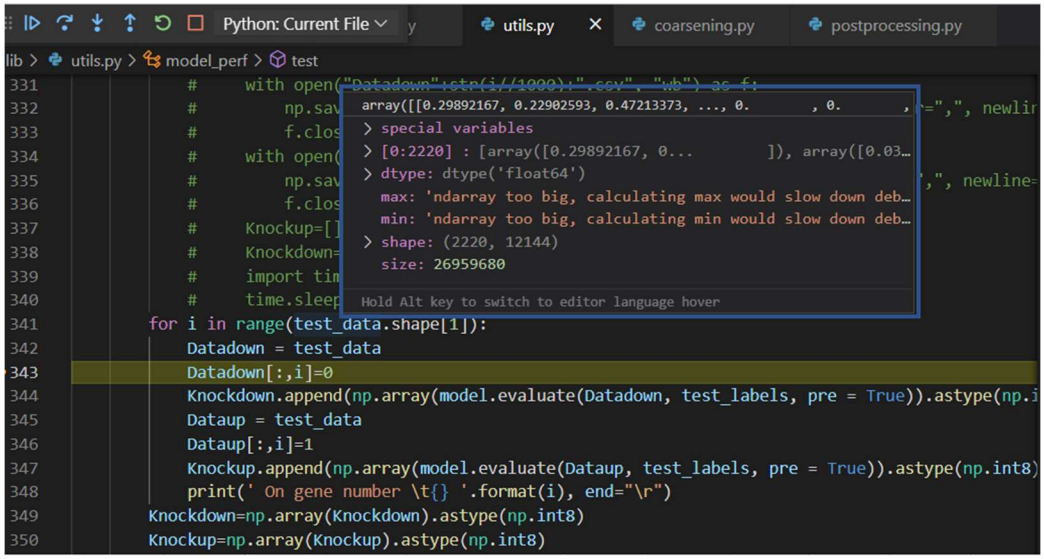


***Figure 27*** *The size of testing data at post-modelling analysis.*

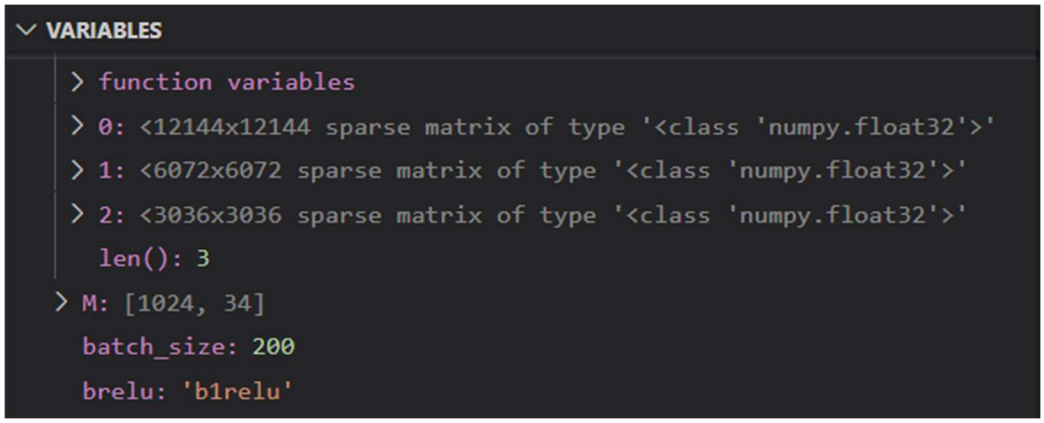


***Figure 28*** *Convolution layers.*

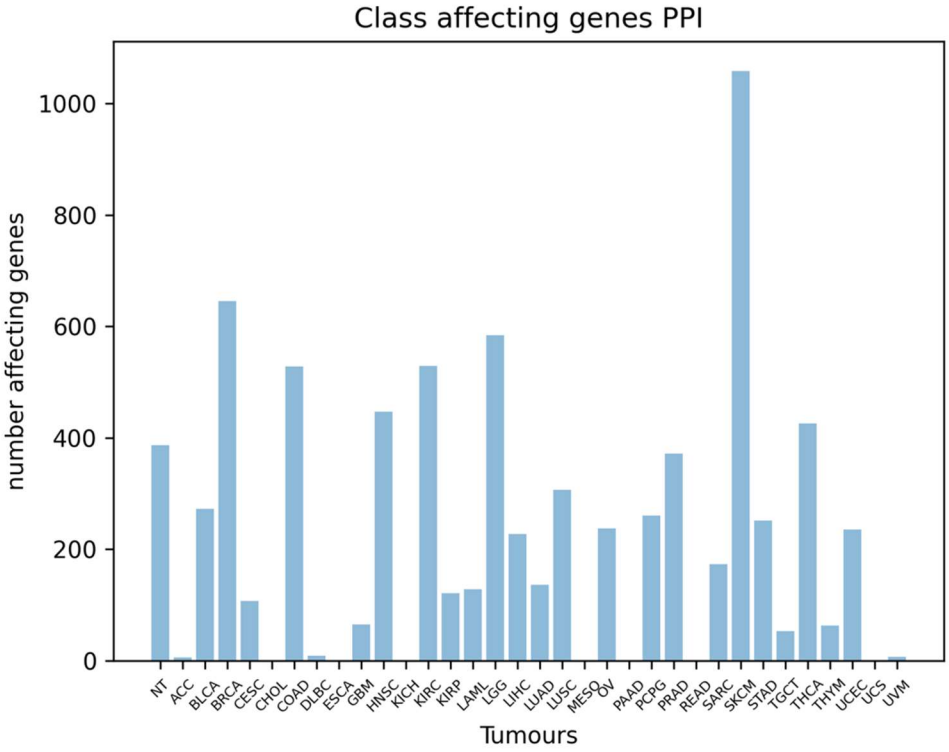


***Figure 29*** *The post-modeling analysis is shown how many time that the result of prediction had been affected by a modification in genes. The data from PPI model.*

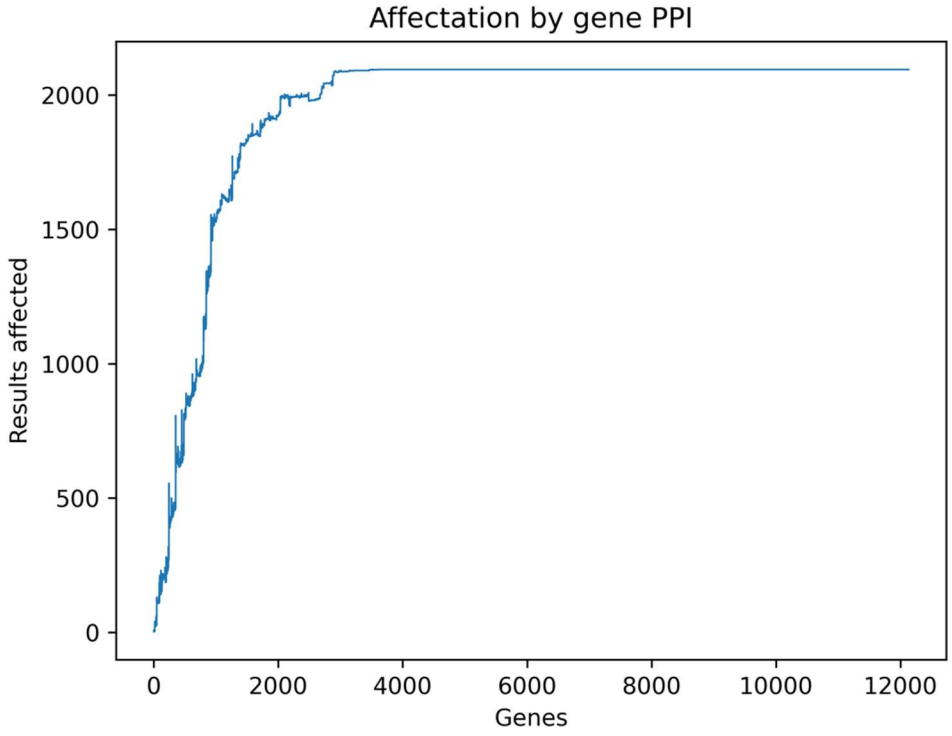


***Figure 30*** *The post-modeling analysis is shown the position of gene which affected the prediction of genes position. The data from PPI model.*

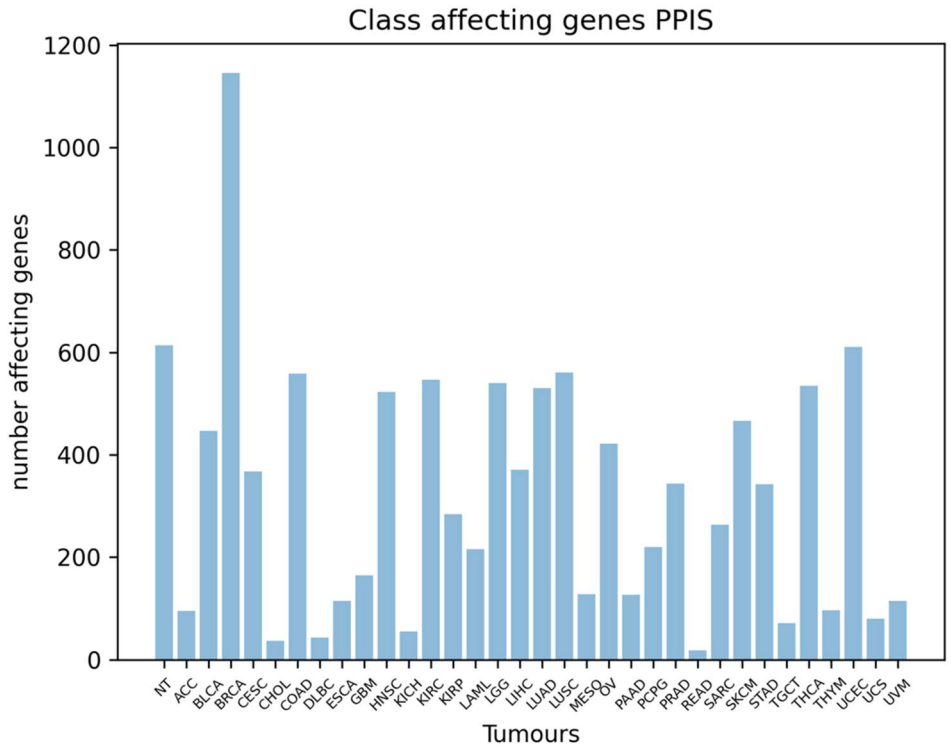


***Figure 31*** *The post-modeling analysis is shown how many time that the result of prediction had been affected by a modification in genes. The data from PPI plus singleton model.*

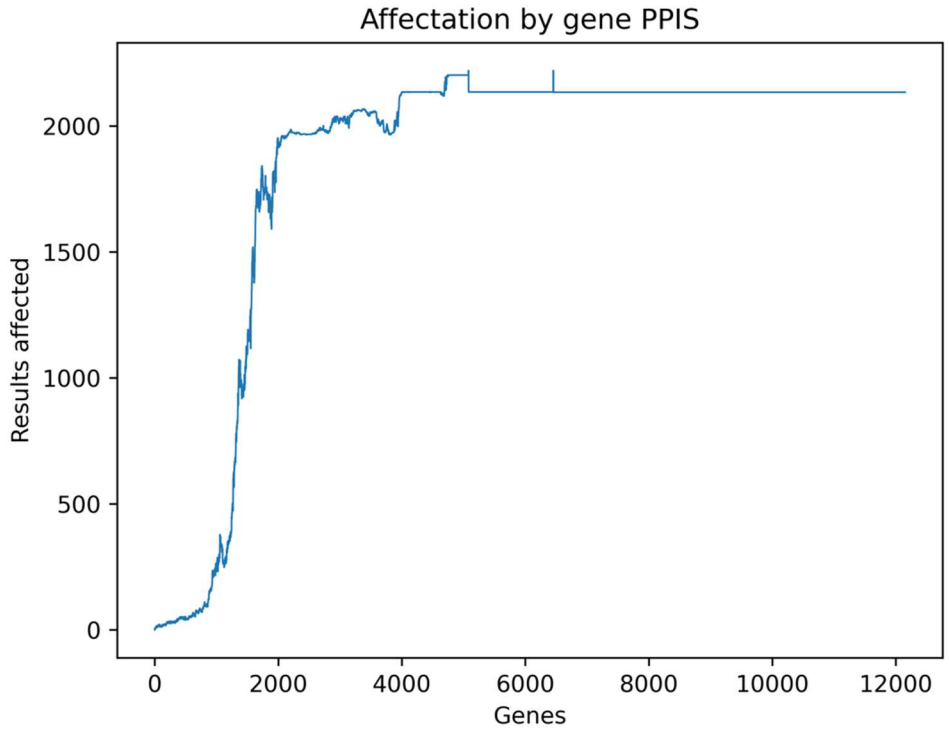


***Figure 32*** *The post-modeling analysis is shown the position of gene which affected the prediction of genes position. The data from PPI plus singleton model.*

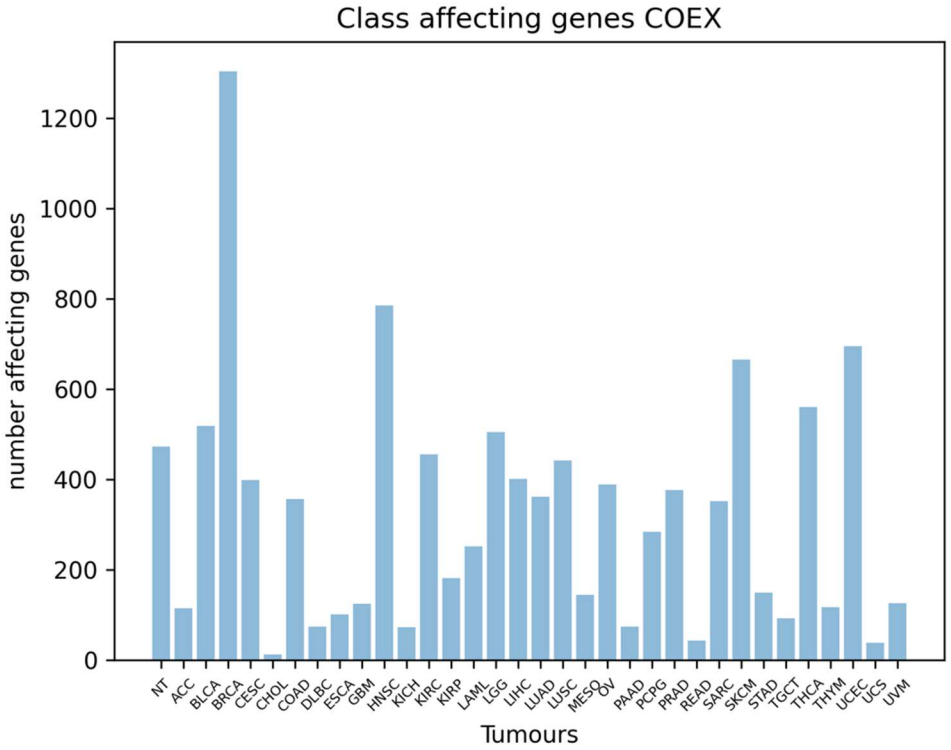


***Figure 33*** *The post-modeling analysis is shown how many time that the result of prediction had been affected by a modification in genes. The data from COEX model.*

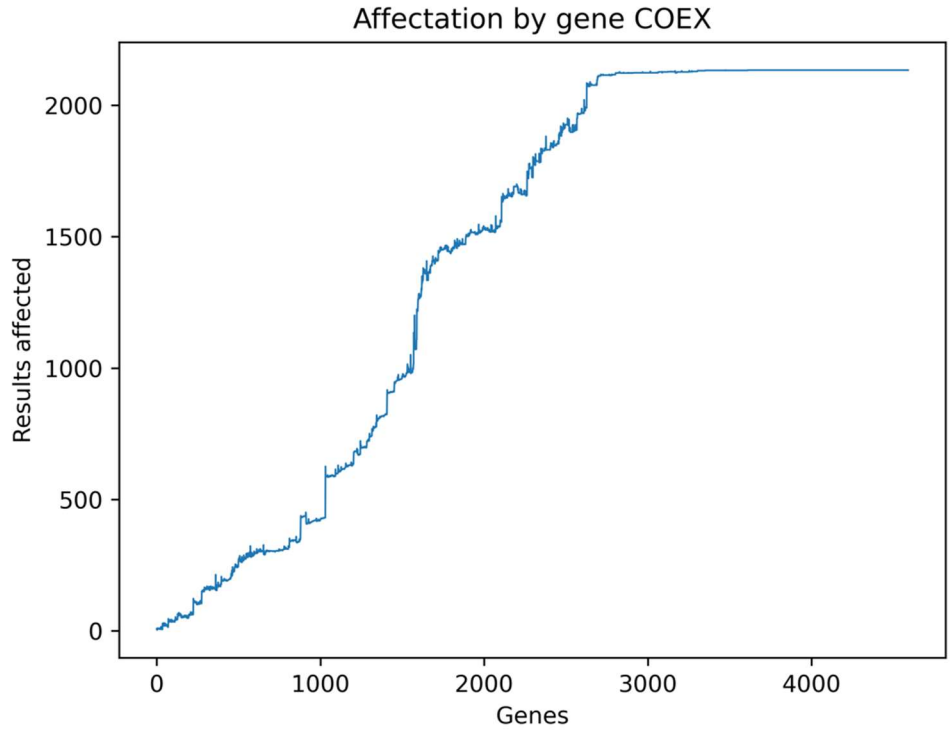


***Figure 34*** *The post-modeling analysis is shown the position of gene which affected the prediction of genes position. The data from COEX model.*

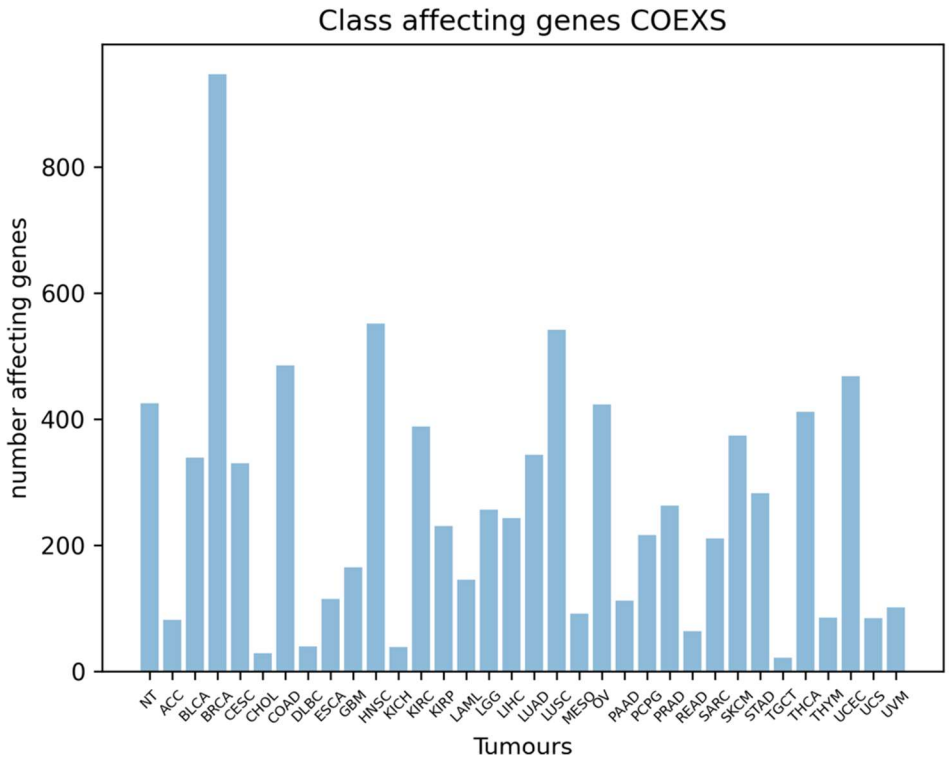


***Figure 35*** *The post-modeling analysis is shown how many time that the result of prediction had been affected by a modification in genes. The data from COEX plus singleton model.*

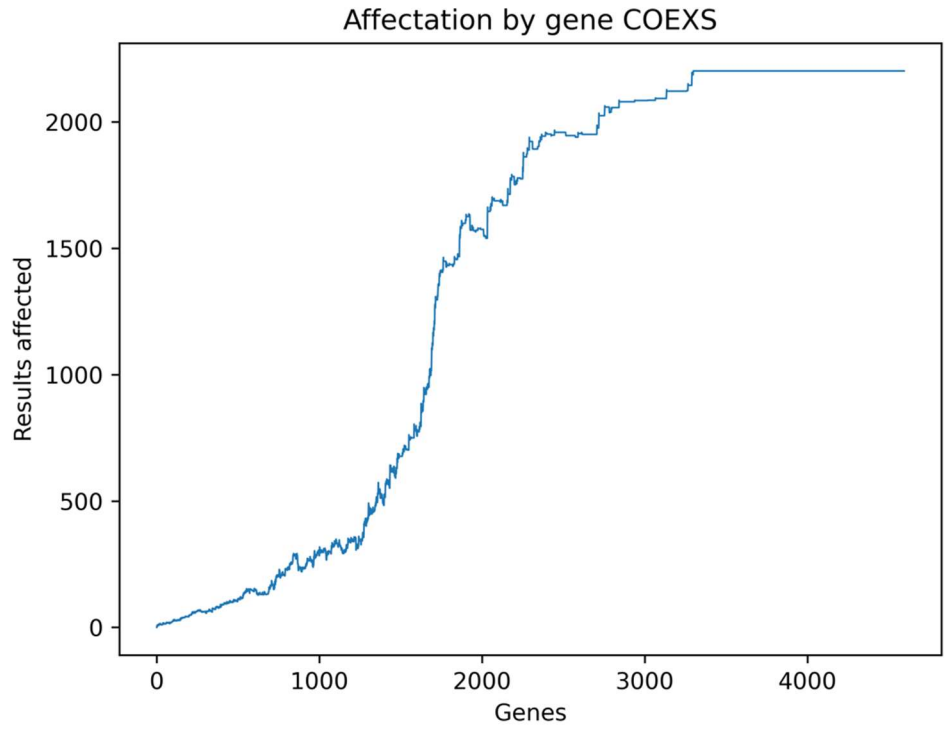


***Figure 36*** *The post-modeling analysis is shown the position of gene which affected the prediction of genes position. The data from COEX plus singleton model.*

## Discussion

Referring to all results of prediction in 32 experiments, each one of experiment was executed 10 times. The totals 320 times were finding the best result of prediction which main objective is to decrease the error of prediction between COAD, READ and STAD, ESCA. The best PPIS GCNN model is 2 coarsening layers, 1024 size of hidden layer and the learning rate is equal to 0.005 in which verify the accuracy of 98.36 as shown in Figure 17. The best PPI GCNN model is 1 coarsening layers, 1024 size of hidden layer and the learning rate is equal to 0.005 in which verify the accuracy of 90.36 as shown in Figure 13. The best COEX GCNN model is 1 coarsening layers, 1024 size of hidden layer and the learning rate is equal to 0.005 in which verify the accuracy of 96.95 as shown in Figure 21. The best COEXS GCNN model is 2 coarsening layers, 1024 size of hidden layer and the learning rate is equal to 0.005 in which verify the accuracy of 96.74 as shown in Figure 25. The number of prediction error of COAD and READ are dropped down from the original GCNN model which the PPIS GCNN model was 158 drops down to 20, the COEX GCNN model was 122 drops down to 43 and the COEXS GCNN model was 100 drops down to 53. The number of prediction error of COAD and READ for PPI GCNN model remains the same from 156 to 143. The number of prediction error of STAD and ESCA are dropped down from the original GCNN model which the PPI GCNN model was 70 drops down to 35, the PPIS GCNN model was 39 drops down to 12, the COEX GCNN model was 37 drops down to 19 and the COEXS GCNN model was 23 drops down to 8. The mean and the standard deviation for all analysis are represented in Table 5. The results derived from "Graph Convolutional Neural Networks applied to classify cancer types" are represented in the 1st results[3]. The 2nd and 3rd are the results from the previous author[4]. The 4th results are derived from this paper which are refined all GCNN model. The mean and the standard deviation of 4th results are PPI is $89.248 \pm 0.51$, PPIS is $96.348 \pm 1.54$, COEX is $95.524 \pm 1.30$ and COEXS is $95.131 \pm 1.44$. The PPI plus singleton GCNN model accomplish the best on average. There is the constant of lowest standard deviation. The post-modeling analysis had predicted how many times of each tumor class is being affected for a gene modification and shown into 2 graphs as shown in Figure 27-28.

| | The 1st results from the classification of cancer types using graph convolution neural networks | The 2nd results from graph convolutional neural networks applied to classify cancer types | The 3rd results using random data 70% train, 15% validation and 15% testing | The 4th results Refinement of a GCNN approach applied to classify cancer types |
|---|---|---|---|---|
| Protein-to-protein interaction | $81.94 \pm 8.66$ | $84.67 \pm 5.37$ | $87.76 \pm 1.85$ | $89.248 \pm 0.51$ |
| Protein-to-protein interaction plus singleton | $93.82 \pm 1.17$ | $93.96 \pm 0.55$ | $93.81 \pm 0.66$ | $96.348 \pm 1.54$ |
| Co-expression | $92.57 \pm 2.73$ | $91.72 \pm 2.68$ | $80.25 \pm 2.68$ | $95.524 \pm 1.30$ |
| Co-expression plus singleton | $93.32 \pm 0.41$ | $93.05 \pm 0.78$ | $93.07 \pm 0.37$ | $95.131 \pm 1.44$ |

***Table 5*** *The comparison of mean and the standard deviation in each GCNN model between previous results and current results.*

## Conclusions

This study thus contributes to refine the graph convolution neural network in order to improve the accuracy of prediction of each cancer types. The four GCNN models were trained by using the data of the cancer tissue as the input to organize the group of the different tumor and the samples of non-tumor into their designated 33 types of the cancers or as normal. The best model of PPI and COEX is one coarsening layer, 1024 size of hidden layer, 200 batch size, 20 epochs and the learning rate is equal to 0.005. The PPI model is improved from (87.76% ± 1.85, mean ± std) to (89.25% ± 0.51, mean ± std). The COEX model is improved from (80.25% ± 2.68, mean ± std) to (95.52% ± 1.30, mean ± std). The best model of PPIS and COEXS is two coarsening layers, 1024 size of hidden layer, 200 batch size, 20 epochs and the learning rate is equal to 0.005. The PPIS model is improved from (93.81% ± 0.66, mean ± std) to (96.35% ± 1.54, mean ± std). The COEX model is improved from (93.07% ± 0.37, mean ± std) to (95.13% ± 1.44, mean ± std). The results show that the main purpose of the master's thesis is to improve the prediction accuracy is achieved. Specifically, to refine the classification of cancer classes that present problems are completed. The PPIS model had original error of COAD and READ equal to 158 errors and currently the error is reduced to 20. The COEXS model had original error of COAD and READ equal to 100 errors and currently the error is reduced to 53. The PPIS model had original error of COAD and READ equal to 156 errors and currently the error is reduced to 143. The COEX model had original error of COAD and READ equal to 122 errors and currently the error is reduced to 43. The prediction error of STAD and ESCA is reduced in four models which are PPI model from 70 to 35, PPIS model from 39 to 12, COEX model from 37 to 19 and COEXS model 23 to 8. Finally, the post-modelling analysis has been reviewed and analysed algorithm leads to more accurate results as showed in results of post-modeling analysis.

## Future work

The future work can proceed this study by testing the new tumor of cancer tissue type particularly of the cancer tissue type which come from similar tissues. As we all know, cancers have many stages so that we should expand the prediction of the same type of cancer but different stage of it. For the technical, the method time is taking 5 to 7 hours including the post-modeling analysis. This program might take time for processing but can be used as diagnostic under the guidance of the physician.

# Specifications of laptop

Figure 37 shows the specifications of the laptop that used in the experiments.

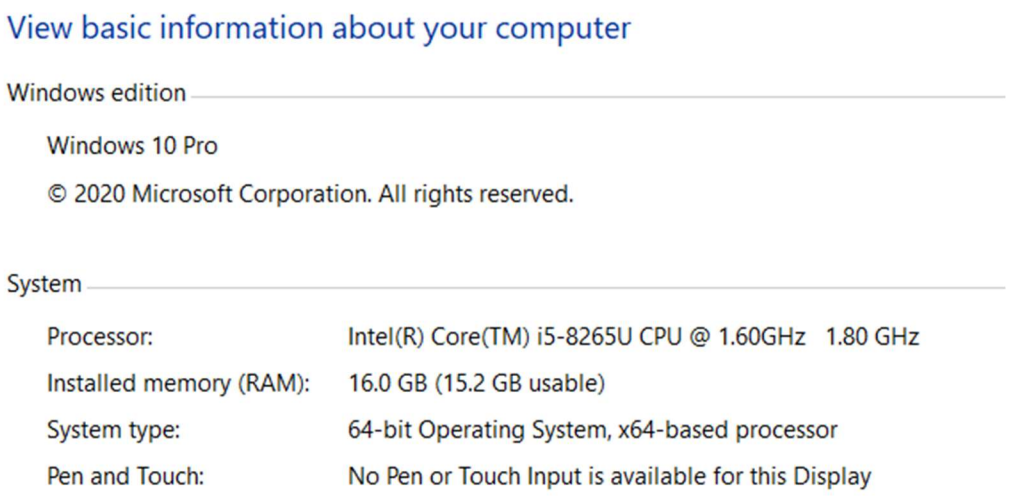


***Figure 37*** *The specifications of the laptop that used in the experiments.*

# References

[1] WHO, Cancer, (2021). https://www.who.int/news-room/fact-sheets/detail/cancer (accessed 23 June 2021)

[2] American Association for Cancer Research(AACR), Cancer Care Costs in the United States Are Projected to Exceed $245 Billion by 2030, (2020). https://www.aacr.org/about-the-aacr/newsroom/news-releases/cancer-care-costs-in-the-united-states-are-projected-to-exceed-245-billion-by-2030/ (accessed 10 June 2020)

[3] R. Ramirez, Y.C. Chiu, A. Hererra, M. Mostavi, J. Ramirez, Y. Chen, Y. Huang, Y.F. Jin, Classification of Cancer Types Using Graph Convolutional Neural

[4] Heribert Pascual Saldaña, Graph Convolutional Neural Networks applied to classify cancer types

[5] Bronstein MM, Bruna J, LeCun Y, Szlam A, Vandergheynst P. Geometric deep learning: going beyond euclidean data. IEEE Signal Proc Mag. (2017) 34:18–42. doi: 10.1109/MSP.2017.2693418

[6] Szklarczyk D, Morris JH, Cook H, Kuhn M, Wyder S, Simonovic M, et al. The STRING database in 2017: quality-controlled protein-protein association networks, made broadly accessible. Nucleic Acids Res. (2017) 45:D362–8. doi: 10.1093/nar/gkw937

[7] Chereda H, Bleckmann A, Kramer F, Leha A, Beissbarth T. Utilizing molecular network information via graph convolutional neural networks to predict metastatic event in breast cancer. Stud Health Technol Inform. (2019) 267:181–6. doi: 10.3233/SHTI190824

[8] Rhee S, Seo S, KimS. Hybrid approach of relation network and localized graph convolutional filtering for breast cancer subtype classification. arXiv preprint arXiv:1711.05859 (2017). doi: 10.24963/ijcai.2018/490

[9] Szklarczyk D, Franceschini A,Wyder S, Forslund K, Heller D, Huerta-Cepas J, et al. STRING v10: protein–protein interaction networks, integrated over the tree of life. Nucleic Acids Res. (2014) 43:D447–52. doi: 10.1093/nar/gku1003

[10] Mostavi M, Chiu Y.-C, Huang Y, Chen Y. Convolutional neural network models for cancer type prediction based on gene expression. BMC Med Genomics. (2020) 13:1–13. doi: 10.1186/s12920-020-0677-2

[11] The Cancer Genome Atlas Program - National Cancer Institute, (n.d.). https://www.cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga (accessed 19 May 2021).

[12] RPKM, FPKM and TPM, clearly explained | RNA-Seq Blog, (n.d.). https://www.rna-seqblog.com/rpkm-fpkm-and-tpm-clearly-explained/ (accessed 19 May 2021).

[13] K.R. Kukurba, S.B. Montgomery, Topic Introduction RNA Sequencing and Analysis, (2015). https://doi.org/10.1101/pdb.top084970.

[14] Szklarczyk D, Morris JH, Cook H, Kuhn M, Wyder S, Simonovic M, et al. The STRING database in 2017: quality-controlled protein-protein association networks, made broadly accessible. Nucleic Acids Res. (2017) 45:D362–8. doi: 10.1093/nar/gkw937

[15] Szklarczyk D, Franceschini A,Wyder S, Forslund K, Heller D, Huerta-Cepas J, et al. STRING v10: protein–protein interaction networks, integrated over the tree of life. Nucleic Acids Res. (2014) 43:D447–52. doi: 10.1093/nar/gku1003

[16] Siska C, Kechris K. Differential correlation for sequencing data. BMC Res Notes. (2017) 10:54. doi: 10.1186/s13104-016-2331-9

[17] Smedley D, Haider S, Durinck S, Pandini L, Provero P, Allen J, et al. The BioMart community portal: an innovative alternative to large, centralized data repositories. Nucleic Acids Res. (2015) 43:W589–8. doi: 10.1093/nar/gkv350

[18] Jeanquartier F, Jean-Quartier C, Holzinger A. Integrated web visualizations for protein-protein interaction databases. BMC Bioinformatics. (2015) 16:195. doi: 10.1186/s12859-015-0615-z

[19] Defferrard M, Bresson X, Vandergheynst P. Convolutional neural networks on graphs with fast localized spectral filtering. Adv Neural Inform Proc Syst. (2016) 3844–52.

[20] Hammond DK, Vandergheynst P, Gribonval R. Wavelets on graphs via spectral graph theory. Appl Comput Harmonic Anal. (2011) 30:129–50. doi: 10.1016/j.acha.2010.04.005

[21] Kipf TN,Welling M. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016).

[22] National Cancer Institute(NIH), Colon Cancer Treatment (PDQ®)–Patient Version, (2021), https://www.cancer.gov/types/colorectal/patient/colon-treatment-pdq#section/all (accessed 3 September 2021)

[23] Wikipedia, Laplacian matrix, (2022). https://en.wikipedia.org/wiki/Laplacian_matrix (accessed 15 January 2022)

[24] Wikipedia, Greedy algorithm, (2019). https://en.wikipedia.org/wiki/Greedy_algorithm (accessed 19 December 2019)

[25] G. Karypis, V. Kumar, Metis: Unstructured Graph Partitioning and Sparse Matrix Ordering, Unstructured Graph Partinioning and Sparse Matrix Ordering. (1995) 1–16.

[26] tf.Graph | TensorFlow Core v2.5.0, (n.d.). https://www.tensorflow.org/api_docs/python/tf/Graph (accessed 26 June 2021).

[27] Introduction to graphs and tf.function | TensorFlow Core, (n.d.). https://www.tensorflow.org/guide/intro_to_graphs (accessed 26 June 2021).

[28] tf.compat.v1.placeholder | TensorFlow Core v2.5.0, (n.d.). https://www.tensorflow.org/api_docs/python/tf/compat/v1/placeholder (accessed 26 June 2021).

[29] numpy.shape — NumPy v1.22.dev0 Manual, (n.d.). https://numpy.org/devdocs/reference/generated/numpy.shape.html (accessed 24 June 2021).

[30] numpy.array — NumPy v1.21 Manual, (n.d.). https://numpy.org/doc/stable/reference/generated/numpy.array.html (accessed 24 June 2021).

[31] numpy.reshape — NumPy v1.21 Manual, (n.d.). https://numpy.org/doc/stable/reference/generated/numpy.reshape.html (accessed 24 June 2021).

[32] numpy.ravel — NumPy v1.21 Manual, (n.d.). https://numpy.org/doc/stable/reference/generated/numpy.ravel.html (accessed 24 June 2021).

[33] GCN Classification: Google Drive, (n.d.). https://drive.google.com/drive/folders/1_Cnvab7mIwCrNJyY-J4aR2ck9i72KH8t (accessed 24 June 2021).

[34] numpy.transpose — NumPy v1.21 Manual, (n.d.). https://numpy.org/doc/stable/reference/generated/numpy.transpose.html (accessed 24 June 2021).

[35] numpy.hstack — NumPy v1.21 Manual, (n.d.). https://numpy.org/doc/stable/reference/generated/numpy.hstack.html (accessed 24 June 2021).

[36] numpy.vstack — NumPy v1.21 Manual, (n.d.). https://numpy.org/doc/stable/reference/generated/numpy.vstack.html (accessed 24 June 2021).

[37] tf.Graph | TensorFlow Core v2.5.0, (n.d.). https://www.tensorflow.org/api_docs/python/tf/Graph (accessed 26 June 2021).

[38] Introduction to graphs and tf.function | TensorFlow Core, (n.d.). https://www.tensorflow.org/guide/intro_to_graphs (accessed 26 June 2021).

[39] tf.compat.v1.placeholder | TensorFlow Core v2.5.0, (n.d.). https://www.tensorflow.org/api_docs/python/tf/compat/v1/placeholder (accessed 26 June 2021).

[40] GitHub - decube83/GCN_Cancer, (n.d.). https://github.com/decube83/GCN_Cancer (accessed 29 July 2021)