ML model for gene expression in Prostate Cancer

A THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR DEGREE OF

MASTER OF TECHNOLOGY
IN
Bioinformatics

By-

Aviral Verma
MBI2020018

under the supervision of

Dr. Ashutosh Mishra

To the:
DEPARTMENT OF APPLIED SCIENCES

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, ALLAHABAD
(A UNIVERSITY ESTABLISHED UNDER SEC.3 OF UGC ACT, 1956 VIDE NOTIFICATION NO. F.9-4/99-U.3 DATED 04.08.2000 OF THE GOVT. OF INDIA)

A CENTRE OF EXCELLENCE IN INFORMATION TECHNOLOGY ESTABLISHED BY GOVT. OF INDIA

MAY 2022

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY

ALLAHABAD

(An Institute of National Importance by the Act of Parliament, Govt. of India)

CANDIDATE DECLARATION

I, **Aviral Verma**, Enrollment No. **MBI2020018**, certify that this thesis work entitled **"ML model for gene expression in Prostate Cancer"** which is submitted by me in partial fulfillment of the requirements for completion of **M.Tech. in Bioinformatics** to to the **Department of Applied Sciences**, **Indian Institute of Information Technology Allahabad**, is an authentic record of work carried out under the guidance of Prof. Ashutosh Mishra and due acknowledgments have been made in the text of the thesis to all the other materials used.

I understand that plagiarism includes:
1. Reproducing someone else's work/ideas (fully or partially) and claiming it as one's own.
2. Reproducing someone else's work (Verbatim copying or paraphrasing) without crediting.
3. Committing literary theft (copying some unique literary construct).

I have given due credit to the original authors/ sources through proper citation for all the words, ideas, diagrams, graphics, computer programs, experiments, results, websites, that are not my original contribution. I have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

Place : **IIIT Allahabad, Prayagraj**

Date: **25/05/2022**

Aviral Verma (MBI2020018)

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY

ALLAHABAD

(An Institute of National Importance by the Act of Parliament, Govt. of India)

CERTIFICATE FROM SUPERVISOR

I do hereby declare that this thesis prepared under my supervision by **Aviral Verma (MBI2020018)** entitled "**ML model for gene expression in Prostate Cancer**" be accepted in partial fulfillment for the degree of **M.Tech in Bioinformatics** for examination.

Date: **25/05/2022**

Place: .**IIIT Allahabad, Prayagraj**

[illegible]

……………………………………………

Dr. Ashutosh Mishra

Thesis Supervisor

Countersigned by:

………………………

Dean(Academics)

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY

ALLAHABAD

(An Institute of National Importance by the Act of Parliament, Govt. of India)

CERTIFICATE OF APPROVAL

The foregoing thesis is hereby approved as a credible study in Bioinformatics and its allied areas. It is carried out and presented in a satisfactory manner to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein but the thesis only for the purpose for which it is submitted.

Committee Members for Evaluation of the Thesis for Final Examination:

[illegible]
....

....

....

....

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY

ALLAHABAD

(An Institute of National Importance by the Act of Parliament, Govt. of India)

Acknowledgments

I'd would like to offer my sincere gratitude to all those who provided me with resources and advice to finish my thesis, special thanks to my classmates **Niraj Raj, Mayank Maravaniya and Jay Vyas.**

I am deeply grateful to **Dr**. **Ashutosh Mishra** for his expert assistance and consistent support in ensuring that this project meets its objectives from start to finish. I am very thankful to him for giving me valuable suggestions and providing me with a better understanding of the topics. His vital contribution to the proposed project has made it possible.

I would like to express my gratitude to all Research Scholars in the Indian Institute of Information Technology Allahabad lab who assisted me with my work.

I am grateful to IIIT Allahabad's Director for offering a research-oriented infrastructure.

Finally, I'd like to extend my sincere thanks to all faculty members who inspired me to complete my work.

ABSTRACT

Cancer diagnosis and treatment requires precise cancer prediction. Important cancer marker genes can be inferred using a predictive model. In the search for cancer genes, One of the crucial steps is identifying the prime regulators and key regulatory pathways.

Prostate cancer is the second to only one in the most common cancers amongst the men that claim the most lives each year around the world. Some prostate tumours spread quickly, but the majority do not. In reality, postmortem investigations demonstrate that moslty older men (and younger men, to some extent) who died due to some other reasons had prostate cancer as well that had never bothered them before. In many instances, neither their doctor nor themselves had the idea that they were suffering from it.

Prostate cancer may begin as a pre-cancerous state, according to some study, albeit this is not proven. When a man undergoes a prostate biopsy, these disorders are sometimes discovered (removing small pieces from the prostate to while looking for cancer). Since data is available, machine learning approaches can help predict prostate cancer at an early stage.

Machine learning is widely employed for a variety of objectives, with several real-world applications. They are utilised in image recognition, speech recognition for robotic analysis, traffic pattern analysis, weather prediction, and other applications. Machine learning offers a number of approaches, strategies, and tools for diagnosing and predicting problems in a variety of medical fields. It has been used to solve fundamental difficulties in genomic medicine and to demonstrate how differences in an individual's DNA can affect their risk of certain diseases.

In this paper, we present a strategy for identifying important regulators in prostate cancer (PCa) using a convolutional neural network built from gene

expression records from PCa patients. Gene expression is normally regulated in a coordinated manner, with one gene's presence or absence affecting the expression of another (gene interaction)

CNN (Convolution Neural Network) is used for gene expression in prostate cancer because of its adaptability, feature extraction, and self-training. Our model has performed well for the dataset. We have a model with better accuracy that takes care of data without computational complexity that can be effectively used for different applications.

Contents

List of Figures

Chapter 1: INTRODUCTION

1.1 About Prostate cancer

Cancer is a spectrum of diseases characterised by the uncontrolled division of aberrant cells that develop and spread uncontrollably throughout the body. Normal cells in the human body create, grow, and divide as they should. Cells are also replaced as they age or get damaged, however tumours are recognized as cells that divide and expand uncontrollably.Cancer arises when the body's cells begin to grow out of control. Cancer cells can form in practically throughout the body parts and spread to other sections of the body. Metastasis is the process through which cancer cells propagate to various areas of the body. It is generally caused by a change in genes, such as mutation in deoxyribonucleic acid (DNA) that is different than normal tissues.

The prostate gland is a gland in the reproductive system of males that out of which seminal fluid is secreted in adult men. Prostate cancer arises when the growth of prostate gland calls start to spiral out of control. In recent years, high-throughput gene expression investigations using techniques including as next generation sequencing, proteomics and microarray have uncovered new genes and pathways in PCa.

The prostate lies beneath the bladder (the vacuous body part that stores urine) and ahead of the rectum (the ending part of the intestines). The majority of the seminal fluid is produced by seminal vesicles, which are located directly behind the prostate. Urethra is the tube that goes through the prostate's centre and takes out the sperm and the urine out of the body through the penis.

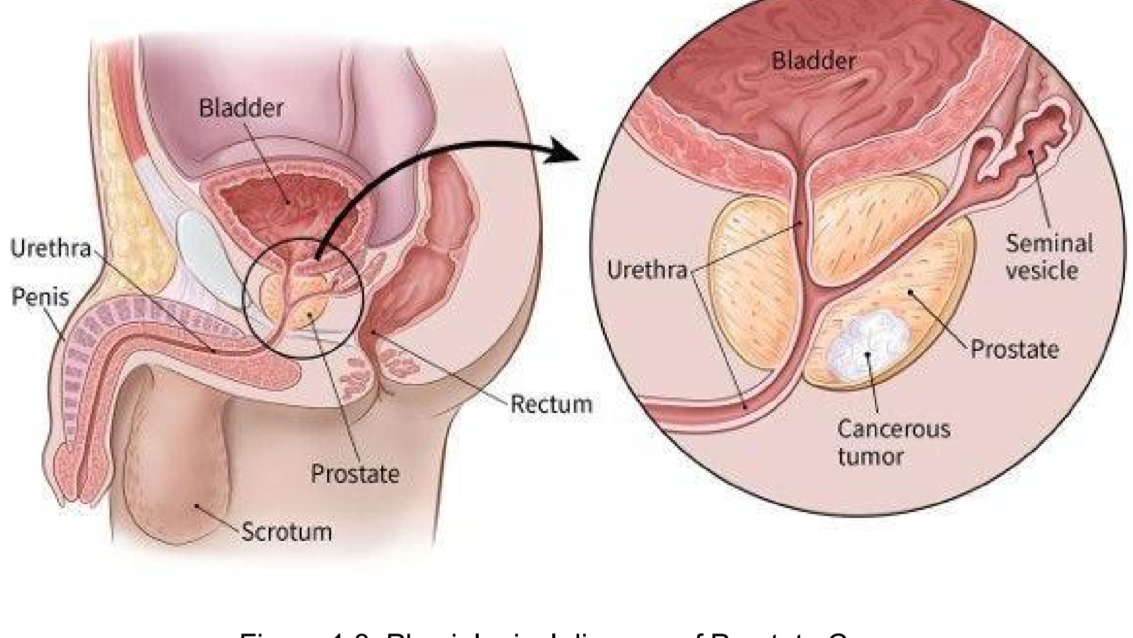


Figure 1.3: Physiological diagram of Prostate Cancer

The size of the prostate changes as a man gains age. In younger men, it is about the dimension of a walnut, but it can be considerably larger in older guys.

Older males are more prone to develop prostate cancer. Men 65 and older account for about 6 out of every 10 instances, while men under 40-45 are rare.The average age of diagnosis for men is 66. Prostate cancer affects about 1 in every 8[1] men at some stage of life. Prostate cancer kills around one guy in every 41. There were over 1.4 million new prostate cancer cases, with an estimated 375,304[2] persons who died due to it.

1.2 Convolution Neural Network

The technique of mixing two functions to produce the output of the other function is known as convolution. A Convolution Neural Network can learn sophisticated objects and patterns thanks to its input layer, numerous hidden layers, output layers and lots of parameters. It subsamples the input using convolution and pooling techniques, applying an activation function, with these partially connected hidden layers , culminating in the output layer.

The convolution neural comes under the domain of deep learning. It is also known as CNN or ConvNet. It is widely used for image classification, identification, image recognition, etc. CNN takes the machine vision and deep learning to new height in research. It takes a image as input and outputs the probability of the classes. In comparison to other recognition and classification algorithms, CNN requires very little preprocessing.It is cost-effective as it does not needs high or complex processing power.

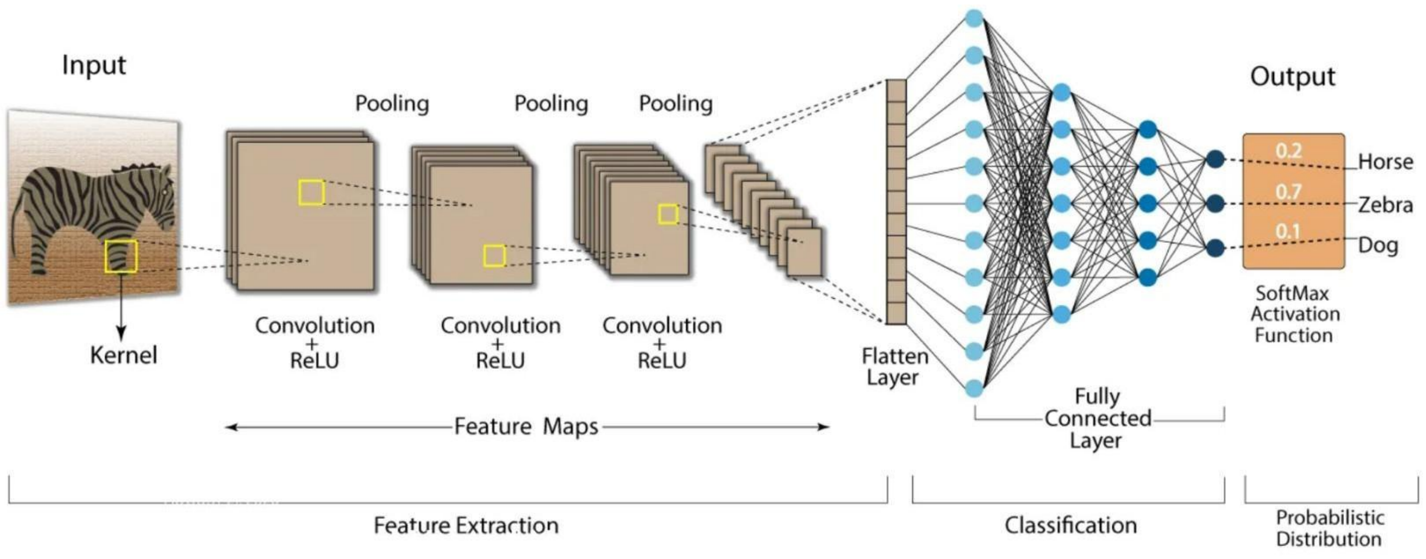


Figure 1.2: CNN Architecture

A convolutional neural network's architecture can be divided into two parts:

1.Feature learning: In this component, CNN learns features from the input such as edges, shades, parts, objects, and soon. It passes the knowledge to the next component after extraction, and

2. Classification: This component generates a probability of class output for prediction

These components can be further divided into three important layers.
Feature learning have:

1) Convolution layer, and
2) Pooling Layer

And Classification layer have:

3) Fully Connected Layer

The convolution layer is a crucial part of CNN. The first layer, the convolution layer, receives the input and extracts the features by conducting component-wise multiplication and then adding with multiple kernels, followed by the ReLU layer. Its more advanced version extracts more complex features such as face parts, objects, forms, and more. This layer outputs a featured matrix, which is then fed into the following pooling layer.

After the convolution layer comes the pooling layer. It takes a feature matrix from the convolution layer as input and outputs a reduced-dimension feature matrix. This layer reduces the dimension to reduce the incoming data's computational complexity. The key features are extracted from the feature matrix. There are two types of feature extraction algorithms: max-pooling and average-pooling. In Max-pooling, the feature with the highest value in comparison to other features in the pooling zone is chosen. Instead of choosing the largest value of features, Average-pooling takes an average of the features present in the pooling zone.

The Fully Connected layer takes the input as a flatten feature matrix and then combines the features to construct the model. Activation functions like the Softmax function are used to generate the output as the likelihood of classes. This output is utilised for classification and recognition. These probability add up to one. This layer is known as a completely linked layer because it functions similarly to a multi-layer Perceptron in that each neuron in one layer is interconnected to each neuron in the next layer.

.

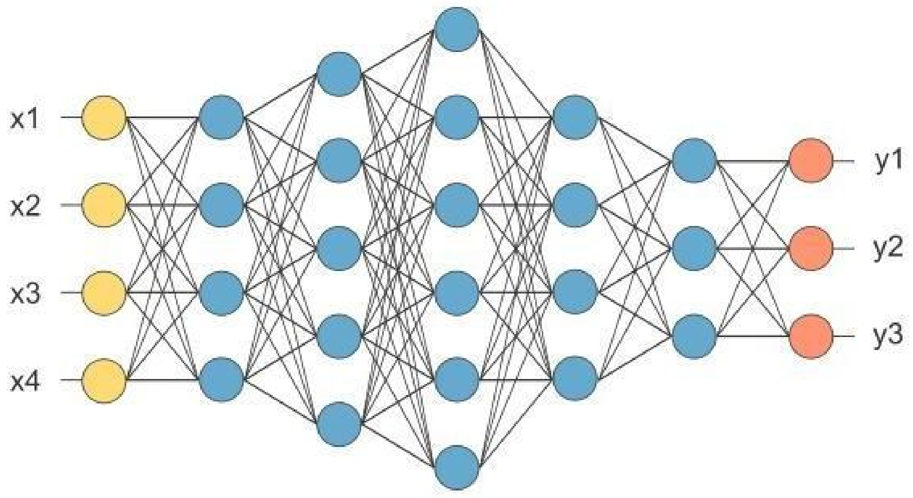


Figure 1.3: Architecture of CNN layers

1.3 Link of the Code

Source Code is present on the following link:

https://github.com/aviralrabbit1/ML-model-for-gene-expression-of-prostate-cancer

1.4 Report's structure

CHAPTER 2: This chapter covers the literature survey on RNA gene expression recognition to acknowledges previous work.
CHAPTER 3: This chapter covers a brief discussion on motivation and objective.
CHAPTER 4: This chapter covers the proposed model and approach with description.
CHAPTER 5: This chapter covers that how we are going to evaluate our model and calculate the accuracy.
CHAPTER 6: This chapter covers the accuracy and results of our classification model.
CHAPTER 7: At last it contains expected future work, conclusion and correlation of our classification model with respect to previous classification models.

Chapter 2: LITERATURE REVIEW

The Cancer Genome Atlas Program[3] is a ground-breaking analysis of cancer gene expression programme that has characterised nearly 20,000 principal cancer and paired regular samples from 33 different cancer types at the molecular level. In 2006, the National Cancer Institute and the National Human Genome Research Institute partnered on this project, which brought together scientists from various fields and institutes. "The results published or reported here are based in part or entirely on data collected by the TCGA Research Network: https://www.cancer.gov/tcga." Over the course of a dozen years, TCGA collected over 2.5 petabytes of genomic, transcriptomic, epigenomic, and proteomic data. The data has already improved our ability to detect, diagnose, and prevent cancer, and will remain open to anybody in the research community.

The study of the transcriptome[4], which comprises all gene readouts presentin a cell, is being revolutionised by RNA-Seq. The human genome is made up ofDNA that contains instructions for constructing and maintaining cells. DNA mustbe read and transcribed or copied into ribonucleic acid to carry out these instructions (RNA). RNA-Seq is a very sensitive and accurate technology for monitoring transcriptome expression that allows researchers to see previously unnoticed alterations that occur in disease situations. Researchers can also detect gene status in cells and tissues using the whole collection of RNA sequences.

They[5] used whole-slide photos from The Cancer Genome Atlas to train a deeplearning model (inception v3) to correctly and autonomously categorize them as LUSC, LUAD, or normal lung tissue in the mentioned paper. OuTher technique's performance is comparable to pathologists', with an estimated area under the curve (AUC) of 0.97. Our model was validated using separate datasets of cryogenic tissues, and biopsies and formalin-fixed paraffin-embedded tissues. The network was also trained to predict the LUAD's 10 most commonly altered genes.

Gene expression is normally regulated in a coordinated manner, with the absence or presence of one gene affecting the expression of some other. Network theory, which uses graphs as representations to analyse interactions between isolated entities, could be applied to the investigation of complex gene regulatory networks (scale-free, random, small world and hierarchical networks). The development of algorithms to study these network could be a useful tool for locating and identifying disease-associated genes in complicated disorders like cancer[6].

Chapter 3: Motivation and Objective

3.1 Motivation

The motivation behind this work is to develop a machine learning model that enhances the prediction of prostate cancer with more accuracy and also handle non-linearity in the data. We are building this model because prostate cancer is the second most life taking cancer type for men today. In this work, I have used the Convolution Neural Network because it can handle the data well. I have used some preprocessing techniques and data augmentation for more accuracy. This proposed work will be very useful for the system which has low computational capabilities but need high accuracy.

3.2 Objective

The main objective of this thesis is to build a ML model which can predict the possibility of prostate cancer with high accuracy by training with a large dataset . And also show how the classification model behaves and accuracy changes for different models using and without using dropout layer.

Chapter 4: PROPOSED METHODOLOGY

In this chapter, we have briefly discussed our methodology and our research work on hand gesture recognition. Here we have proposed an architecture and in the next chapter, we have done evaluation and analyzed the result. Our proposed architecture can be divided into two components:

Model development component is divided into four parts:

1. Data Collection
2. Data Preprocessing
3. CNN model development, and
4. CNN model testing and evaluation.

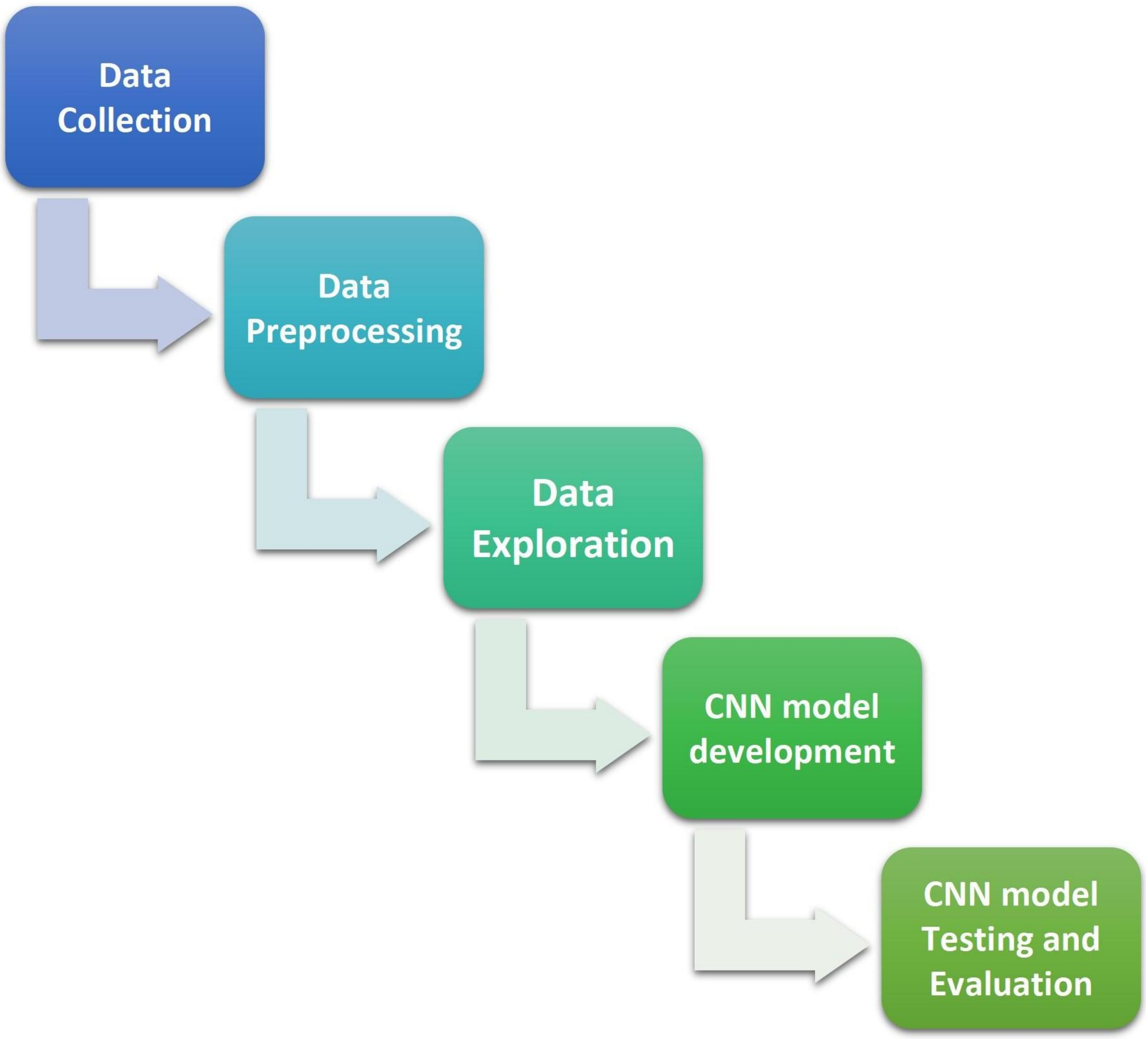
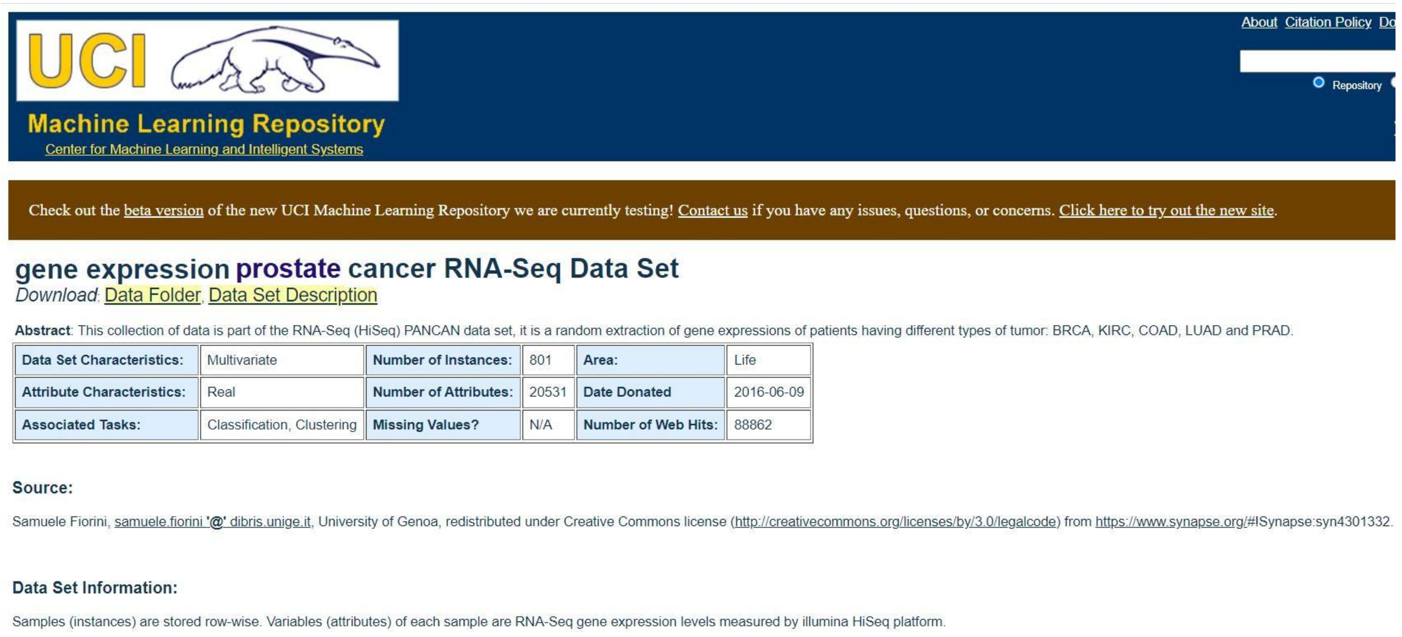


Fig 4.1 Overview of Model Development

4.1 Data Collection

The dataset is taken from UC Irvine Machine Learning Repository[7], acquired from The Cancer Genome Atlas (TCGA), which is the world's largest database of genomic information. (TCGA) is a collaborative effort between the National Human Genome Research Institute and the National Cancer Institute's Center for Cancer Genomics that began in 2006 [NCI]. The data has improved everything from appropriate diagnosis to treatment and prevention of prostate cancer. Many researchers have found it much easier to conduct study because data is open source. There are 20531 attributes in the TCGA dataset used in this thesis, each with 801 Instances. This TCGA dataset was used to create our model.


About Citation Policy Do
Repository
UCI
Machine Learning Repository
Center for Machine Learning and Intelligent Systems

Check out the beta version of the new UCI Machine Learning Repository we are currently testing! Contact us if you have any issues, questions, or concerns. Click here to try out the new site.

gene expression prostate cancer RNA-Seq Data Set
Download: Data Folder, Data Set Description

Abstract: This collection of data is part of the RNA-Seq (HiSeq) PANCAN data set, it is a random extraction of gene expressions of patients having different types of tumor: BRCA, KIRC, COAD, LUAD and PRAD.

Data Set Characteristics:	Multivariate	**Number of Instances:**	801	**Area:**	Life
Attribute Characteristics:	Real	**Number of Attributes:**	20531	**Date Donated**	2016-06-09
Associated Tasks:	Classification, Clustering	**Missing Values?**	N/A	**Number of Web Hits:**	88862

Source:

Samuele Fiorini, samuele.fiorini '@' dibris.unige.it, University of Genoa, redistributed under Creative Commons license (http://creativecommons.org/licenses/by/3.0/legalcode) from https://www.synapse.org/#!Synapse:syn4301332.

Data Set Information:

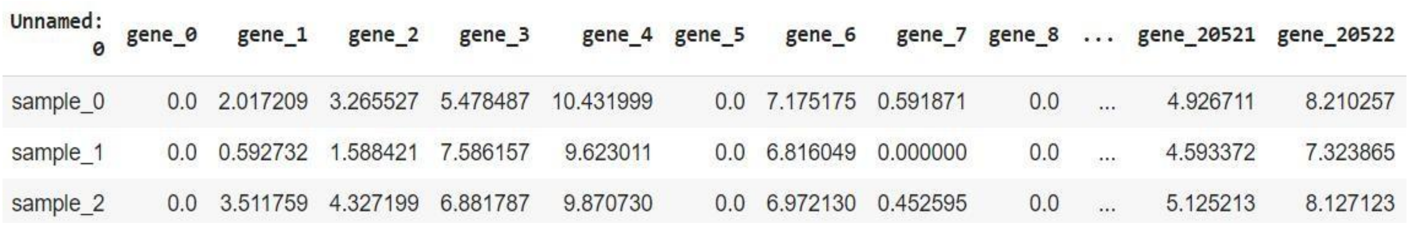
Samples (instances) are stored row-wise. Variables (attributes) of each sample are RNA-Seq gene expression levels measured by illumina HiSeq platform.

Fig 4.2 UCI - ML repository

4.2 Data Preprocessing

In machine learning and deep learning, data preprocessing is essential. it allows us to convert raw data into usable formats or extract the most important characteristic from the data for our classification model. It also aids in improving model accuracy by extracting relevant features and reducing model computing complexity by reducing the dimension.

Fortunately, the data I got from the repository has already been preprocessed (from RNA sequenced probed mircroarray data) and converted into suitable form of data (Luminescence values of the respective gene expression).



Unnamed: 0	gene_0	gene_1	gene_2	gene_3	gene_4	gene_5	gene_6	gene_7	gene_8	...	gene_20521	gene_20522
sample_0	0.0	2.017209	3.265527	5.478487	10.431999	0.0	7.175175	0.591871	0.0	...	4.926711	8.210257
sample_1	0.0	0.592732	1.588421	7.586157	9.623011	0.0	6.816049	0.000000	0.0	...	4.593372	7.323865
sample_2	0.0	3.511759	4.327199	6.881787	9.870730	0.0	6.972130	0.452595	0.0	...	5.125213	8.127123

Fig 4.3 Luminescence values of genes

4.3 Data Exploration

❖ We have overall 801 classes in the data, out of which, five are unique, namely -

- ➢ BRCA
- ➢ COAD
- ➢ KIRC
- ➢ LUAD
- ➢ PRAD

❖ For using these classes in building our model, we will convert these classes into numerical values.

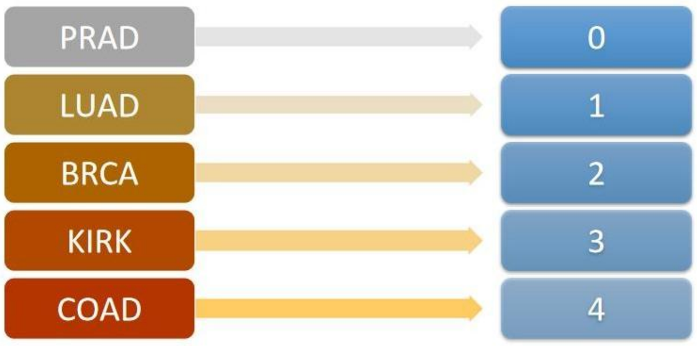


Fig 4.5 Unique Classes in the data, converted to numeric values

❖ For building the model, we merge both the tables (containing the luminescence values of the person/sample, and the labels of cancer pertaining to each of these samples).

❖ The data is then normalized.

I have split the data into 60:20:20 - training data, testing data and validation data.

- ➢ 60% - training set
- ➢ 20% - validation set, and
- ➢ 20% - testing set (prediction and matching)

4.4 Model Development

For my planned project, I have used CNN. It contains various layers for doing various tasks. I settled at this layer configuration after experimenting with a huge number of distinct layer combinations. Some layers are utilised for feature extraction, while others are employed to handle over-fitting and categorization. This model receives the luminescence values as input and returns the accuracy of the predictions as an output.

I have employed -

- **tf.keras.Sequential** - suitable for a simplistic stack layer with only single input tensor alongwith single output tensor on every layer.

- **tf.keras.layers.Dense** - Normal well-connected layer of Neural Network.

  output = activation(dot(input, kernel) + bias)

- **tf.keras.layers.Dropout** - helps to reduce overfitting by randomly setting input units to 0 with some frequency rate at every step duringin the training period

- **Rectified Linear Unit (ReLU)** - A simple linear function which directly returns the input if it is positive; else, it returns zero. Oftenly Gives superior performance while it is easy to train .

- **Softmax** - Mathematical function which can convert a integer vector into a probability vector, with the posibility of each value corresponding to the relative scale of vector.

- **Categorical_crossentropy** - This loss function is utilised when a multi-class classifier model has 2 or more labels of output. The output label receives a single-shot category encoding value in the form of 0s and 1s.Keras converts the output label from integer to categorical encoding if it is in integer form.

- **Adam** - An optimizer, one of the two arguments [compile() & fit()] required for compiling a Keras mode.

- **Metrics** - ‘accuracy’, used to judge the performance of your mode.

Chapter 5. EVALUATION AND RESULTS

I built and initial model (without the dropout layer, thus carrying the risk of overfitting) and I built another, final model (with the dropout layer, thus without the risk of overfitting) and I calculate the loss and accuracy for each one of them which I have noted below.
For the below provided plots-

- blue = accuracy/Loss
- red = val_accuracy/Val_loss

For model 1(without dropout):

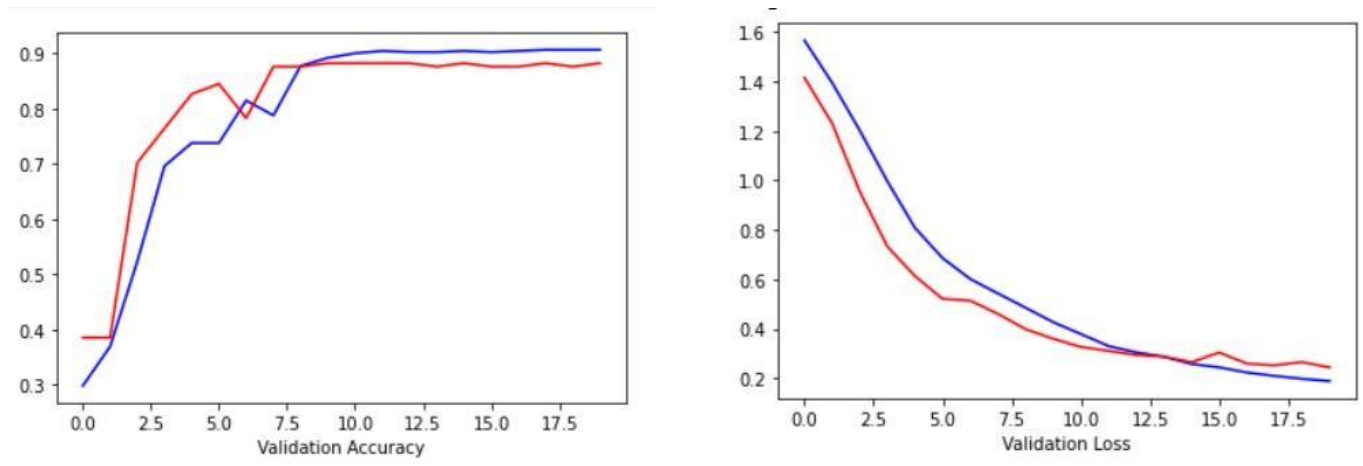


Fig 5.1 (a) Model 1's validation accuracy and (b) validation loss

The loss and accuracy of the first model comes out to be:

- loss = 0.2432
- accuracy ≈ 0.8820

For model 2(with dropout):

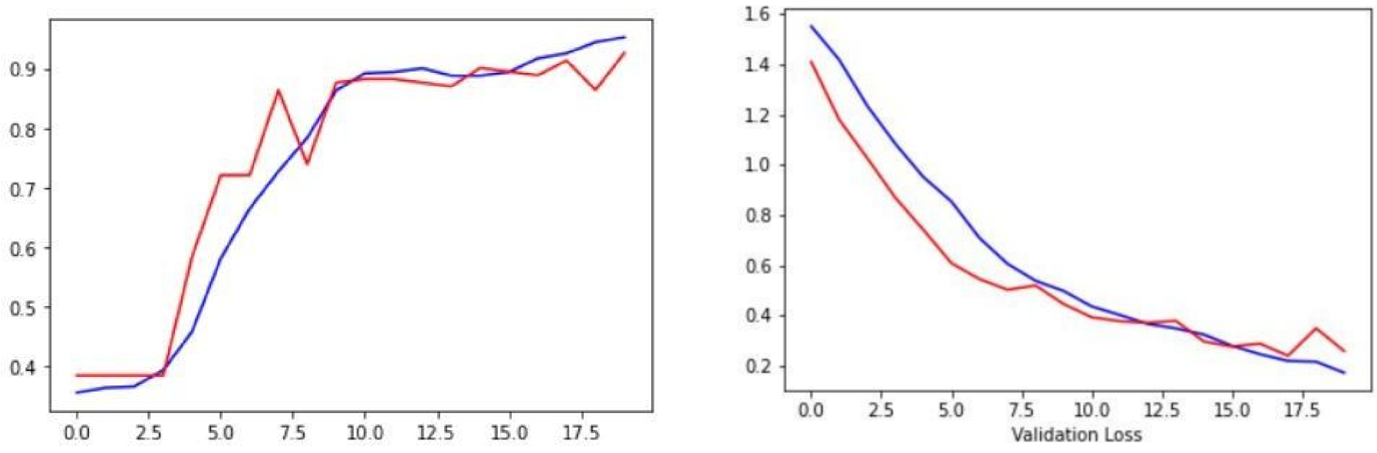


Fig 5.2 (a) Model 2's validation accuracy and (b) validation loss

	CNN(without dropout)	CNN(with dropout)
Loss	0.2432	0.2597
Accuracy	0.8820	0.9255

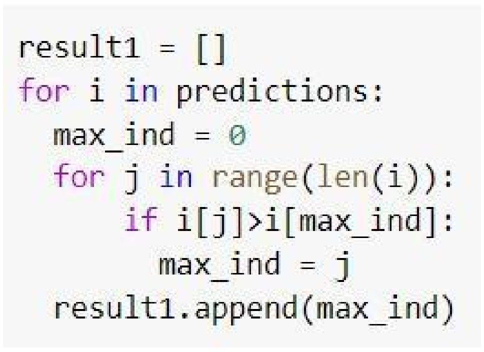
Fig 5.3 Table comparing validation accuracy and validation loss
For both the models

The loss and accuracy of the final model comes out to be:

- loss = 0.2597
- accuracy ≈ 0.9255

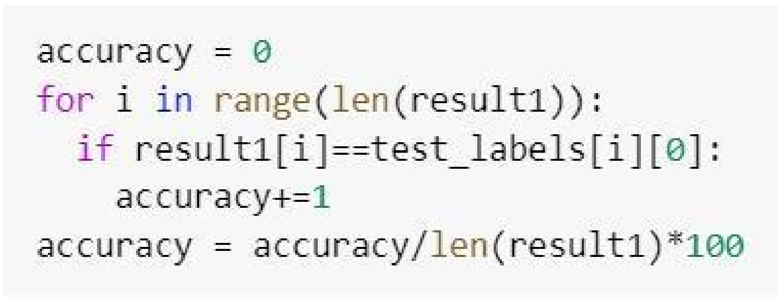
I made predictions using my train model and matched them with the testing set that was spared till now. The accuracy of the match between predicted set and testing set came out to be-
accuracy = **86.875**

The formula/codes I have used to get this accuracy are given below-



```
result1 = []
for i in predictions:
  max_ind = 0
  for j in range(len(i)):
      if i[j]>i[max_ind]:
        max_ind = j
  result1.append(max_ind)
```

Fig 5.4 Code for predicting results/class



```
accuracy = 0
for i in range(len(result1)):
  if result1[i]==test_labels[i][0]:
    accuracy+=1
accuracy = accuracy/len(result1)*100
```

Fig 5.5 Code for calculating accuracy

Chapter 6. Conclusion and Future work

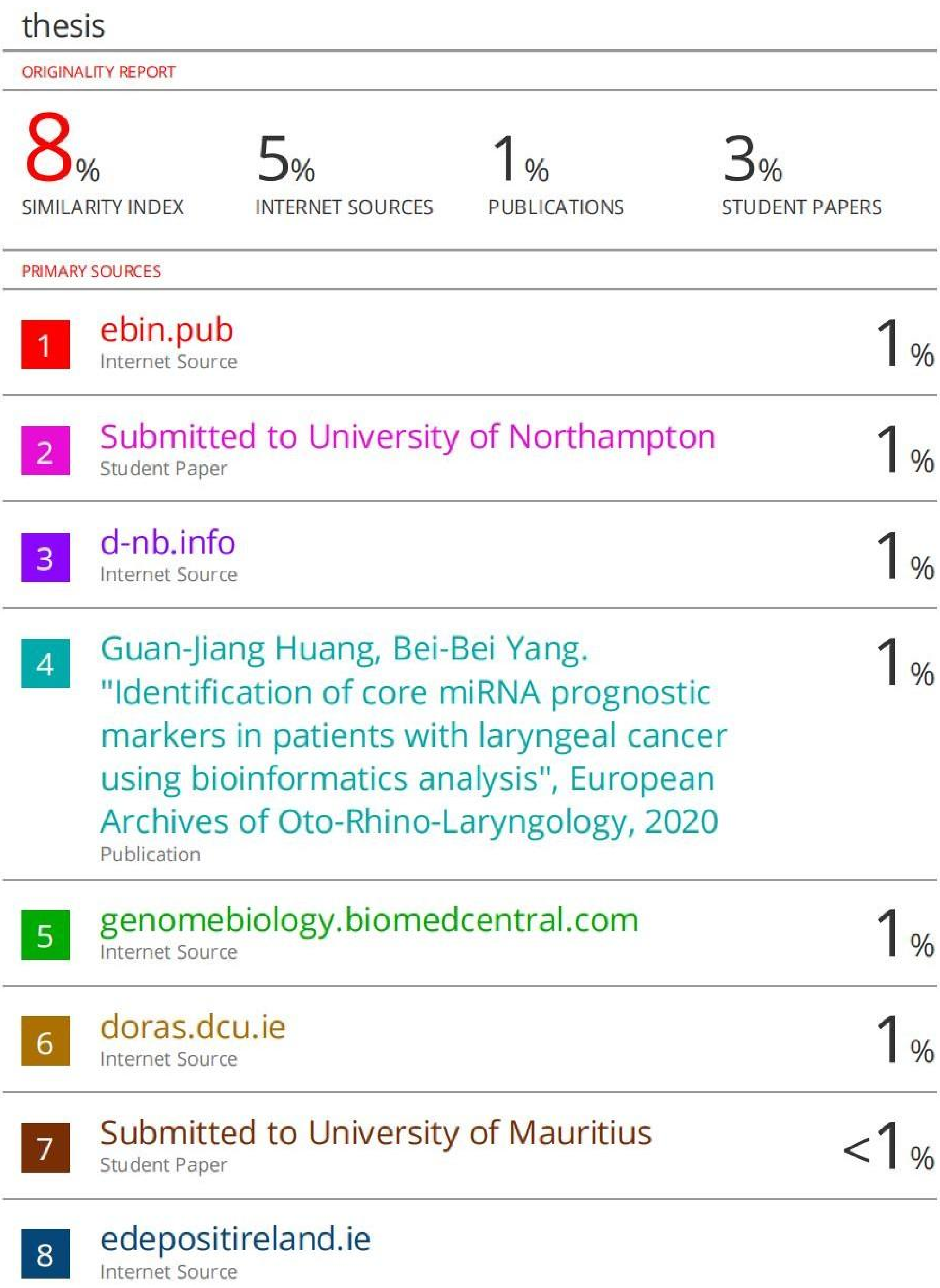
6.1 Conclusion

I have demonstrated how to handle non-linear datasets and how to achieve maximum accuracy on the CNN model in this thesis. We've also discussed about how accuracy differs before and after adding the dropout layer to control the overfitting. We may also deduce from the results that CNN performs better when the size is smaller.

6.1 Future work

In the future, we can expect to see even better machine learning models built on evenlarger amount of dataset with more and more features to reach even higher precision levels.
There could be comparative study for different models like ANN, SVM for work in similar domain/field eg. Lung Cancer, Colon cancer, etc.
We can analyze the effect of the project in real life and thus, say with confidence that there is large scope for developing finer tuned models that will help the future generations only in better and more accurate prediction of cancer and to live a better life through such achievments.

Chapter 7. Plagiarism Report



thesis

ORIGINALITY REPORT

8%	5%	1%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	ebin.pub Internet Source	1%
2	Submitted to University of Northampton Student Paper	1%
3	d-nb.info Internet Source	1%
4	Guan-Jiang Huang, Bei-Bei Yang. "Identification of core miRNA prognostic markers in patients with laryngeal cancer using bioinformatics analysis", European Archives of Oto-Rhino-Laryngology, 2020 Publication	1%
5	genomebiology.biomedcentral.com Internet Source	1%
6	doras.dcu.ie Internet Source	1%
7	Submitted to University of Mauritius Student Paper	<1%
8	edepositireland.ie Internet Source	

		<1 %
9	Submitted to Higher College of Technology Student Paper	<1 %
10	Submitted to Indian Institute of Information Technology, Design and Manufacturing - Kancheepuram Student Paper	<1 %
11	Submitted to University of Birmingham Student Paper	<1 %
12	www.jpbms.info Internet Source	<1 %
13	www.dovepress.com Internet Source	<1 %
14	library2.smu.ca Internet Source	<1 %
15	uwspace.uwaterloo.ca Internet Source	<1 %

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		

References

[1] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6497009/

[2] https://www.cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga

[3] Mangangcha, I.R., Malik, M.Z., Küçük, Ö. et al. Identification of key regulators in prostate cancer from gene expression datasets of patients. Sci Rep 9, 16420 (2019). https://doi.org/10.1038/s41598-019-52896-x

[4] Krishanpal Anamika. Transcriptomic Profiling Using Next Generation Sequencing - Advances, Advantages, and Challenges. page Ch. 4. IntechOpen, Rijeka, 2016.

[5] Coudray, N., Ocampo, P.S., Sakellaropoulos, T. et al. Classification and mutation prediction from non–small cell lung cancer histopathology images using deep learning. Nat Med 24, 1559–1567 (2018). https://doi.org/10.1038/s41591-018-0177-5

[6] Maharjan, Aashi, "Machine Learning Approach for Predicting Cancer Using Gene Expression" (2020). UNLV Theses, Dissertations, Professional Papers, and Capstones. 3922.

[5] The Cancer Genome Atlas Research Network., Weinstein, J., Collisson, E. et al. The Cancer Genome Atlas Pan-Cancer analysis project. Nat Genet 45, 1113–1120 (2013). https://doi.org/10.1038/ng.2764

[6] John N. Weinstein, Eric A. Collisson, Gordon B. Mills, Kenna M. Shaw, Brad A. Ozenberger, Kyle Ellrott, Ilya Shmulevich, Chris Sander, Joshua M. Stuart,1 and Cancer Genome Atlas Research Network.

[7] Tomlins, S.A. et al. Science 310, 644–648 (200).

Appendices

- Libraries and modules to be used:
  - import tensorflow as tf
  - import numpy as np
  - import pandas as pd
  - import os
  - import matplotlib.pyplot as plt
  - from tensorflow.keras.utils import to_categorical   #Converts a class vector (integers) to binary class matrix.
  - from tensorflow.keras.backend import one_hot   #Computes the one-hot representation of an integer tensor
  - from sklearn.model_selection import train_test_split

- data = pd.read_csv('drive/MyDrive/Gene/data.csv',delimiter = ',')
- labels = pd.read_csv('drive/MyDrive/Gene/labels.csv',delimiter = ',')

- data.describe()
- labels.describe()

- data.head()
- labels.head()

- Checking if data has null values
  - data.isnull()

labels['Unnamed: 0']
labels['Class']

- Changing the datatype of the labels class to numeric values so we can use one_hot encoder or to-categorical.
  - labels = labels.replace({'PRAD': 0})
  - labels = labels.replace({'LUAD': 1})
  - labels = labels.replace({'BRCA': 2})
  - labels = labels.replace({'KIRC': 3})
  - labels = labels.replace({'COAD': 4})

  - labels['Class'].unique() - array([0, 1, 2, 3, 4])

df = pd.DataFrame(data)

- Removing the unnecessary row and merging both the data and the labels-
  - df = pd.merge(df, labels, left_on='Unnamed: 0', right_on='Unnamed: 0', how='left').drop('Unnamed: 0', axis=1)

- Splitting the dataset into Testing, Training and validation set
  - ➢ from sklearn.model_selection import train_test_split
  - ➢ train, test = train_test_split(df, test_size = 0.4, random_state=42, shuffle = True)

  - ➢ from pathlib import Path
  - ➢ train_path = os.path.join('drive/MyDrive/Gene/split60_40','train.csv')
  - ➢ test_path = os.path.join('drive/MyDrive/Gene/split60_40','test.csv')
  - ➢ train.to_csv(train_path, sep=',', index= False)
  - ➢ test.to_csv(test_path, sep=',', index= False)
  - ➢ test, valid = train_test_split(test, test_size = 0.5,random_state=42,shuffle = True)
  - ➢ test_path = os.path.join('drive/MyDrive/Gene/split60_40','test.csv')
  - ➢ valid_path = os.path.join('drive/MyDrive/Gene/split60_40','valid.csv')
  - ➢ test.to_csv(train_path, sep=',', index= False)
  - ➢ valid.to_csv(valid_path, sep=',', index= False)

  - ➢ train_labels = train['Class']
  del train['Class']
  train_features = train

  - ➢ test_labels = test['Class']
  del test['Class']
  test_features = test

  - ➢ valid_labels = valid['Class']
  del valid['Class']
  valid_features = valid

- To make the in layers into categorical data, we will use the util module from keras but first we convert it into nummpy array with pandas-
  - ➢ train_labels = pd.DataFrame(train_labels).to_numpy()
  - ➢ test_labels = pd.DataFrame(test_labels).to_numpy()
  - ➢ valid_labels = pd.DataFrame(valid_labels).to_numpy()
  - ➢ train_features = pd.DataFrame(train_features).to_numpy()
  - ➢ test_features = pd.DataFrame(test_features).to_numpy()
  - ➢ valid_features = pd.DataFrame(valid_features).to_numpy()

- to_categorical module need numeric value; so we might have to convert all this into numeric values
  - ➢ train_labels_categorical = to_categorical(train_labels,5)

- We have a lot of features- 20531 features so we need to one hot encode our labels and check how many unique values are there in pandas to decide that many categories-
  - ➢ train_labels_one_hot = tf.keras.backend.one_hot(train_labels, 5)

- We will normalize with tensorflow numpy array along the vertical axis -for each gene , first running vertically downwards across rows (axis 0), and the second running horizontally across columns (axis 1).
  - train_feature_norm = tf.keras.utils.normalize(train_features, axis=0, order=2)
  - test_feature_norm = tf.keras.utils.normalize(test_features, axis=0, order=2)
  - valid_feature_norm = tf.keras.utils.normalize(valid_features, axis=0, order=2)

Model building

- Initial Model

```
def model_1():
➢ model = tf.keras.Sequential([
➢ tf.keras.layers.Dense(5,activation='relu',input_shape=(20531,),name='dense1_5')
                        tf.keras.layers.Dense(16,activation='relu',name='dense5_16'),
                        tf.keras.layers.Dense(32,activation='relu',name='dense16_32'),
                        tf.keras.layers.Dense(5,activation='softmax',name='last')
  ])

model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy']
)
 # We can choose a wide variety of optimizer and loss
  model.summary()
  return model

➢ checkpoint_path = 'drive/MyDrive/Gene/model_1/saved'
➢ cp_callback =
  tf.keras.callbacks.ModelCheckpoint(checkpoint_path,monitor='val_loss',save_be
  st_only=True,save_weights_only=False)

➢ history_1 =
  model_1.fit(train_feature_norm,train_labels_categorical,epochs=20,validation_da
  ta=(valid_feature_norm,valid_labels_categorical),callbacks=[cp_callback])
➢ plot(history_1)

➢ loss1 , accuracy1 =
  model_1.evaluate(valid_feature_norm,valid_labels_categorical)
➢ print(loss1 , accuracy1)
```

- Final Model

```
➢ def model_2():
  model = tf.keras.Sequential([

tf.keras.layers.Dense(5,activation='relu',input_shape=(20531,),name='dense1_5'),
                        tf.keras.layers.Dense(32,activation='relu',name='dense5_32'),
                        tf.keras.layers.Dropout(0.5,name='dropout'),
                        tf.keras.layers.Dense(64,activation='relu',name='dense32_64'),
                        tf.keras.layers.Dense(5,activation='softmax',name='last')
  ])
```

- model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
- model.summary()
- return model
  - #os.mkdir('drive/MyDrive/Gene/model_2')
  - #os.mkdir('drive/MyDrive/Gene/model_2/saved')
  - os.listdir('drive/MyDrive/Gene')

  - checkpoint_path = 'drive/MyDrive/Gene/model_2/saved/cp-{epoch:04d}.ckpt'
  - checkpoint_dir = os.path.dirname(checkpoint_path)
  - cp_callback = tf.keras.callbacks.ModelCheckpoint(checkpoint_path,monitor='val_loss',save_best_only=False,save_weights_only=False)
  - model_2 = model_2()

  - history_m2 = model_2.fit(train_feature_norm,train_labels_categorical,epochs=20,validation_data=(valid_feature_norm,valid_labels_categorical),callbacks=[cp_callback])
  - plot(history_m2)
  - loss_m2 , accuracy_m2 = model_2.evaluate(valid_feature_norm,valid_labels_categorical)
  - print(loss_m2 , accuracy_m2)

- Prediction and matching-

```
➢ predictions = model_1.predict(test_feature_norm)
➢ Predictions
➢ result1 = []
➢ for i in predictions:
➢   max_ind = 0
➢   for j in range(len(i)):
➢       if i[j]>i[max_ind]:
➢         max_ind = j
➢   result1.append(max_ind)

➢ result1

➢ accuracy = 0
➢ for i in range(len(result1)):
➢   if result1[i]==test_labels[i][0]:
➢     accuracy+=1
➢ accuracy = accuracy/len(result1)*100

➢ accuracy
```