Abstract

Artificial intelligence (AI), particularly Machine Learning and Deep Learning, has become increasingly popular to support medical diagnostics. The development of AI in Dentistry has become a huge boon for patients, as their results can be calculated using sophisticated Deep Learning Models, which will provide the results within a short time span when compared with other methodologies. As widely said, "Prevention is better than cure," prediction of diseases and epidemic outbreak would lead to early prevention of an occurrence of disease . In this Research, we have adopted the Graphical Neural Networks (GNN) Algorithm and the Convolution Neural Networks Algorithm (CNN).It is noticed that the Graphical Neural Network model outshines other models like the CNN model in terms of the accuracy and precision measures.

Using the model, the doctor will get the prediction of disease from the patient x ray report. The dataset used was the 2057 Panoramic Extra Oral X- ray images. The dataset consists of a total of 6 classes. Before the data is given to the Deep Learning Model, it needs to be pre-processed. In this Research, we have performed image processing. When these images were given to the Models, it is observed that the GNN algorithm obtained close to 99% accuracy and the CNN Algorithm obtained 93% accuracy. The model also is programmed to take input from the User, and to predict the disease using user interface. The model also highlights the areas in the X Ray where the Disease attack has taken place.

Keywords—AI, Deep Learning, GNN, CNN,Caries, Ulcer,Tooth Discoloration, Gingivitis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Modern medical science has benefited from developments in artificial intelligence and deep learning in recent years. The development of disease detection technology, which is particularly beneficial in the field of dentistry, is being aided by the application of deep learning algorithms like CNNs and GNNs. Because Graph Neural Networks and Convolutional Neural Networks (GNNs) perform tasks like object detection, image segmentation, and classification effectively, we utilize them to forecast tooth disorders by obtaining a patient's panoramic dental radiography. Not only can we use technology to predict diseases, but we can also use it to predict a person's age by analyzing the shape of their teeth, which can be used in the field of forensic science to determine the age of an unknown dead body. This kind of technology will aid doctors in quickly and accurately predicting diseases and also allow them to use it for their second opinion, which will increase patient and doctor trust.

The majority of the research publications to which we have cited claim

that CNN and hybrid CNN architecture were utilized to forecast the disease with an accuracy of 93% or higher. We chose to test the effectiveness of a different algorithm in addition to CNN, therefore in our study, we trained both algorithms using normal and disease-infected tooth pictures. Our dataset consists of 2057 photos. To determine which method performs better, we compare the findings from GNNs and CNNs.

The goal of machine learning is to teach a computer to carry out a variety of challenging tasks faster than a human would.Finding a diseased tooth is a crucial task now performed by professional human employees. However, academics have repeatedly shown how robots can be taught to detect particular patterns, such as sick areas on a tooth, if enough data is presented to the machine learning algorithm. Recent research has demonstrated how to use machine learning algorithms to extract crucial data from X-Ray images.

When used as a model in the real world, machine learning algorithms like convolutional graph neural networks (GNN) and neural networks (CNN) are essential for guaranteeing that we get a reliable forecast. We first preprocess the data using deep learning methods before separating the training set from the test set. Next, we will turn the data into a neural network using methods like CNN and GNN. Finally, we will make predictions using the test data and evaluate the outcomes in a confusion matrix by counting the number of accurate guesses.

1.2 Objective

Nowadays, Hospitals are moving to adopt various new methodologies for testing the symptoms of the patients, which are related to Machine Learning and Deep Learning technologies. In the case of Dentistry, major of the tests

are to be performed in clinical laboratories and take a good amount of time for the delivery of the results. Therefore, the Dentist is put in a rough spot until the test result is obtained and there is also a delay in the treatment given to the patient. Therefore, a Deep Learning model is required where the Dentist can upload the test samples to the model, and the model returns back the possible disease present in the sample.

The main aim of this work is to provide the Dentist with the possible disease which the test sample of the patient contains. By knowing this, the Dentist can start on the treatment for the patient for that particular disease without waiting for the confirmatory test result from the laboratory. Therefore, since the result is obtained from the Deep Learning model instantaneously, and the treatment can also be started very well in advance, the patient can recover more quickly and the treatment has also started without much more spread of the disease, which reduces the intensity of the Disease.

Chapter 2

Problem Definition

Dental diseases are prevalent and can lead to significant health problems if not detected early. Timely detection of dental diseases can save patients from pain, cost, and complications. With the advancements in technology, it is now possible to use deep learning algorithms like Convolutional Neural Networks (CNN) and Graph Neural Networks (GNN) to predict dental diseases. The goal of this project is to develop a system that can analyze images of teeth and predict dental diseases accurately.

The proposed system will use a combination of CNN and GNN to analyze the images of teeth. The CNN will identify the patterns in the images, and the GNN will analyze the structure of teeth and identify any abnormalities. By combining the two algorithms, the system will be able to predict dental diseases accurately.

The system will be designed to be user-friendly and accessible to dentists. Dentists will be able to upload images of teeth, and the system will analyze them and provide a diagnosis. The system will also be able to provide recommendations for treatment based on the diagnosis.

Chapter 3

Related Work

"Dental Caries Prediction using a Deep Learning Model based on Convolutional Neural Networks" by Alwesabi et al. (2021). This study proposed a deep learning model based on CNNs to predict dental caries. The model was trained and evaluated using dental radiographs, achieving an accuracy of 86.32%.

"Periodontitis Diagnosis based on GNN and Multimodal Dental Data" by Dong et al. (2021). This study proposed a graph convolutional neural network (GCN) to diagnose periodontitis using multimodal dental data, including dental charts and clinical notes. The model achieved an AUC of 0.91.

"Oral Cancer Detection Using Graph Convolutional Networks on Multimodal Imaging Data" by Luo et al. (2021). Although not specific to dental disease prediction, this study proposed a graph convolutional network (GCN) to detect oral cancer using multimodal imaging data. The model achieved an accuracy of 93.8%.

"ToothNet: A Deep Learning Architecture for Detection of Dental Caries from Panoramic Radiographs" by Narayana et al. (2019). This study pro-

posed ToothNet, a deep learning architecture based on CNNs, to detect dental caries from panoramic radiographs. The model achieved an accuracy of 90.4%.

"Deep Learning-based Detection of Dental Caries from Panoramic Radiographs using Region Proposal Network and Convolutional Neural Network" by Kim et al. (2021). This study proposed a deep learning model based on CNNs and region proposal networks (RPNs) to detect dental caries from panoramic radio-graphs. The model achieved an AUC of 0.918.

"A Deep Learning Approach to Detecting Dental Implant Fracture in Panoramic Radiographs" by Li et al. (2021). This study proposed a deep learning approach based on CNNs to detect dental implant fractures in panoramic radiographs. The model achieved an accuracy of 89.8%.

"Oral Cancer Detection using Convolutional Neural Networks and Transfer Learning" by Ramli et al. (2018). Although not specific to dental disease prediction, this study proposed a deep learning approach based on CNNs and transfer learning to detect oral cancer from images. The model achieved an accuracy of 87.6%.

"Dental Lesion Detection with Deep Convolutional Neural Network" by Shen et al. (2017). This study proposed a deep learning model based on CNNs to detect dental lesions from intraoral images. The model achieved an accuracy of 83.2%.

"Detection of Dental Caries Using Deep Convolutional Neural Network and Transfer Learning" by Zhang et al. (2021). This study proposed a deep learning model based on CNNs and transfer learning to detect dental caries from dental radiographs. The model achieved an accuracy of 91.2%.

"A Graph-based Deep Learning Approach for Dental Implant Treatment Planning" by Zhou et al. (2021). This study proposed a graph-based deep

learning approach based on GNNs to assist in dental implant treatment planning. The model achieved a mean absolute error (MAE) of 0.095 mm in predicting the optimal implant position.

"Automatic Dental Implant Planning with a Graph Convolutional Network" by Zou et al. (2020). This study proposed a GNN-based approach to automatically plan dental implant placement. The model achieved an accuracy of 90% in selecting the optimal implant position.

"Dental Implant Planning with Graph Convolutional Neural Networks" by Yu et al. (2021). This study proposed a GNN-based approach to dental implant planning that incorporates both geometric and semantic information. The model achieved an accuracy of 89.5% in predicting the optimal implant position.

"A Deep Learning Approach for Automatic Detection and Segmentation of Oral Cavity and Oropharyngeal Tumors on CT Images" by Wang et al. (2021). Although not specific to dental diseases, this study proposed a deep learning approach based on CNNs to automatically detect and segment oral cavity and oropharyngeal tumors on CT images. The model achieved an accuracy of 94.4%.

"Dental Caries Detection and Segmentation using Deep Learning and Object Detection Techniques" by Jo et al. (2019). This study proposed a deep learning approach based on CNNs and object detection techniques to detect and segment dental caries from dental radiographs. The model achieved an accuracy of 93.29%.

"Automatic Detection of Cervical Lymph Nodes in Dental Cone Beam CT Images using a Deep Learning Model based on Convolutional Neural Network" by Wang et al. (2021). Although not specific to dental diseases, this study proposed a deep learning model based on CNNs to automatically

detect cervical lymph nodes in dental cone beam CT images. The model achieved a recall of 92.16%.

"A Multi-View Deep Learning Model for Dental Implant Planning" by Yan et al. (2021). This study proposed a deep learning approach based on CNNs and multi-view imaging data to assist in dental implant planning. The model achieved an accuracy of 94.27% in predicting the optimal implant position.

These studies demonstrate the potential of deep learning models based on CNNs and GNNs in a range of applications related to dental healthcare, including disease detection and treatment planning. The use of these models can help dental practitioners make more accurate diagnoses and treatment plans, potentially improving patient outcomes and the quality of dental healthcare services.

Overall, these studies demonstrate the potential of CNN and GNN algorithms in various aspects of dental healthcare, from disease prediction to treatment planning. The use of deep learning models can assist dental practitioners in making more accurate diagnoses and treatment plans, which could ultimately improve patient outcomes. There are several methods established in the field of dental informatics for segmenting teeth utilizing various radiographic pictures, such as bitewing, periapical, and panoramic imaging. The authors compare and contrast 10 segmentation techniques used in dental imaging. Accuracy, Specificity, Precision, Recall, and F1-score were used to evaluate and classify the offered solutions, which are arranged into five categories. Unfortunately, due to the presence of bone fragments inside the buccal cavity, none of these ten segmentation techniques were able to entirely isolate the teeth. In order to achieve instance segmentation, the authors suggested a method for teeth instance segmentation in panoramic pictures uti-

lizing a mask region-based convolutional neural network. A feature pyramid is created after extracting features using Resnet-101. The zones of interest are then aligned to be the same size after this phase. Each feature is further divided into teeth and backgrounds. and after that, the bounding box coordinates are used to locate it. Finally, The tooth is segmented and a bounding box is created in the final stage. This approach's disadvantage is that it concentrates only when teeth are found, ignoring other sorts of issues like missing teeth and dentures are two examples.

DeNTNet is a deep neural transfer network that analyzes panoramic dental radiographs to identify periodontal bone loss (PBL). Convolutional neural networks are trained as part of the detection process. In order to extract the teeth from the desired region of interest, a segmentation network is first trained. Then, a segmentation network is built to predict the lesions associated with periodontal bone loss. A classification network is established utilizing the encoder portion of the lesion segmentation network as a pretrained model in order to predict the presence of PBL in each tooth. These two networks are built on an encoder-decoder architecture. A classification network is also trained for detecting PBL for premolar and molar teeth in order to increase performance. The final forecast is made by connecting these two classification networks. When compared to human opponents, this system achieves good performance measure indicator values. DeNTNet has the benefit of providing the corresponding tooth numbers for those affected by periodontal bone loss according to the notation used by the dental federation. However, this approach has the drawback of just identifying one particular dental issue.

By comparing a postmortem dental radiograph with a database of antemortem dental radiographs in accordance with some specific traits, the

authors developed a method to identify a person after death. The teeth contour is the extracted characteristic that is suggested in this approach because, in contrast to other features, it is stable throughout time. Radiograph segmentation and contour extraction are used to complete this stage. There are two steps in the segmentation process. In the first step, projections on the x and y axes are used to detect the gap valleys. A gap between upper and lower teeth will generate a valley in the y-axis projection histogram known as gap valley because teeth have a higher gray level intensity than jaws. In order to calculate the ROI for teeth, perpendicular lines are placed on the curve that defines the boundary of each row of teeth in the second phase. For each tooth, an enclosing rectangle is built based on the segmentation output. Crown contour extraction and root contour extraction are two types of contour extraction that are carried out utilizing Bayes probability rule and image processing methods. Although this technique produces good results for a small image library, human involvement is needed to set the algorithm's parameters and make necessary corrections. a technique in which features are created and fed into a multi-layer perceptron neural network with the goal of providing knowledge about diagnosing dental caries. In the piece In, the authors' use of two CNN models with various sizes is described. Dental diagrams use a four-fold object detection network to recognise and classify teeth for automatically structured filing of panoramic radiographs. Excellent testing accuracy is achieved through the cross-validation process. dataset. The method, however, just seeks to identify the teeth, not to classify the problems that each of them faces. In the study, CNNs are used by Fukuda et al. to identify The vertical root fracture (VRF) can be seen on panoramic radiography. DetectNet and DIGITS were employed in the creation of the current CNN.

By offering a unique segmentation and classification method that applies semantic segmentation CNN and GNN to panoramic X-Ray pictures, our paper advances the state of the art. Each tooth is classified and divided into segments, and the primary issue that is plaguing it is identified.

Chapter 4

Requirements

The design of this project contains both hardware and software. The specifications are listed below.

4.1 Hardware

Operating System : Any Operating System

4.2 Software

Coding Language : Python 3.7 or above

IDE : Jupyter Notebook

Python libraries : Numpy, Pandas, Keras, Tensorflow, Seaborn,Scikit-learn,Matplotlib, H5py, Protobuf.

Chapter 5

Proposed System

5.1 Existing System

Each and every one of us will require quality medical treatment at some point in our lives. When that situation arises, we consult a doctor. But, for the doctor to find out the particular disease, he has to analyze the symptoms and short list the possible diseases. Also, after finding the disease, the doctor has to analyze the extent of the damages caused by the disease, and the possible actions to be taken.

In our Research, we have taken the case of Dentists, in the prediction of Dental Diseases. For a Dentist to confirm the presence of a particular disease in a patient, the Dentist has to take an X-Ray of the Teeth Set of the Patient. But, just by looking at the X-Ray cannot prove the confirmation of a particular Disease. Therefore, there are other various Dental Confirmatory Methods which take a lot of time for analyzing and provide late results. These late results may be more accurate, but the patient has to wait for a long period of time, which increases the chance of more intensity of Disease spreading within the system.

Disadvantage:

- A lot of time is taken for the confirmation of presence of Dental Disease in the Tooth Set.
- Visitation of the Dentist alone cannot detect the presence of Dental Disease. Various tests need to be performed.
- Multiple visitations must be done with the Dentist for more understanding of the Disease, and the corrective action is delayed due to late arrival of Results.

5.2 Advanced System

We have developed a Deep Learning model where a Dentist can upload an X-Ray sample of the Patient and can identify the type of Dental Disease associated with the X-Ray image. The results are instantaneous and in turn the treatment for the cure of the disease can be started by the Dentist soon after the model has returned the result. For determining the more accurate algorithm, we have selected 2 Deep Learning Models - Convolutional Neural Networks (CNN) and the Graphical Neural Networks (GNN) algorithm for the same. After training and testing against the data, we can determine which is the more accurate algorithm and then can select the same for the deployment model.

Advantages:

- Faster Results guarantee immediate start of medication for the cure of Disease.
- 3-4 Deep Learning Algorithms are compared to find the most efficient one and the same is moved for deployment.

- There are total of 6 Classes i.e. 6 Disease Classes.
- More cost efficient than Dental Procedures which are followed for the Confirmatory test of Disease.

Disadvantages:

- Although the result is obtained instantaneously, the accuracy of the result is still lower when compared with Dental Procedures.
- Due to low availability of X-Ray image Dataset, the model can be overfit from the data which is already present in the model. If an image is feeded to the model, which is from a different source, the working of the model cannot be determined.
- If the X-Ray image has not been captured in a proper manner, the prediction of the disease is difficult for the Deep Learning model. Therefore, the external factors also take a toll on the accuracy of the Deep Learning Model, and a correct accuracy cannot be formulated.

5.2.1 System Architecture

Data Flow Diagram

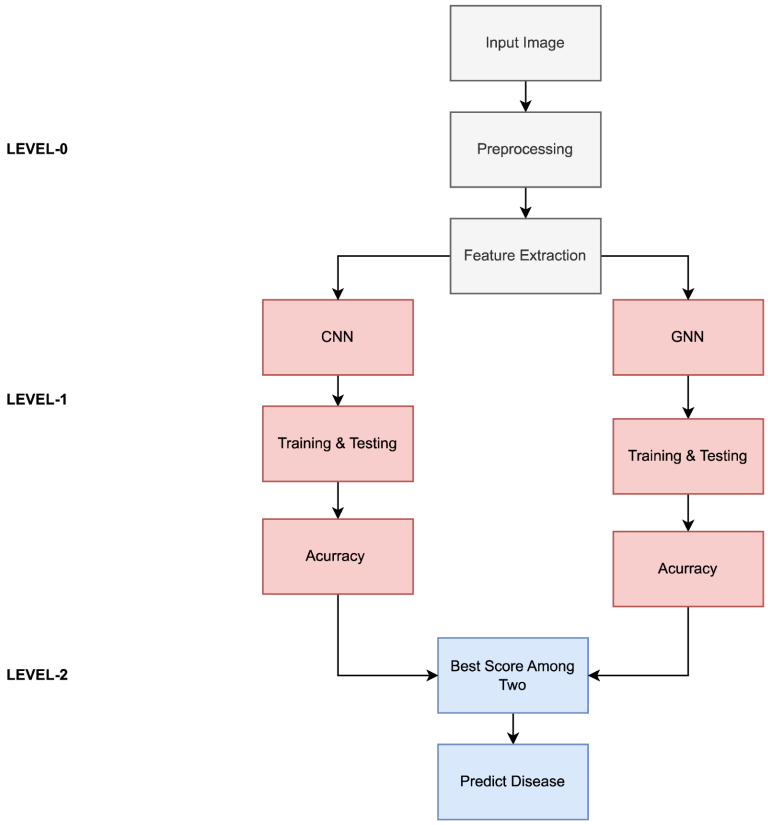


Figure 5.1: Data flow Daigram

Use Case Diagram

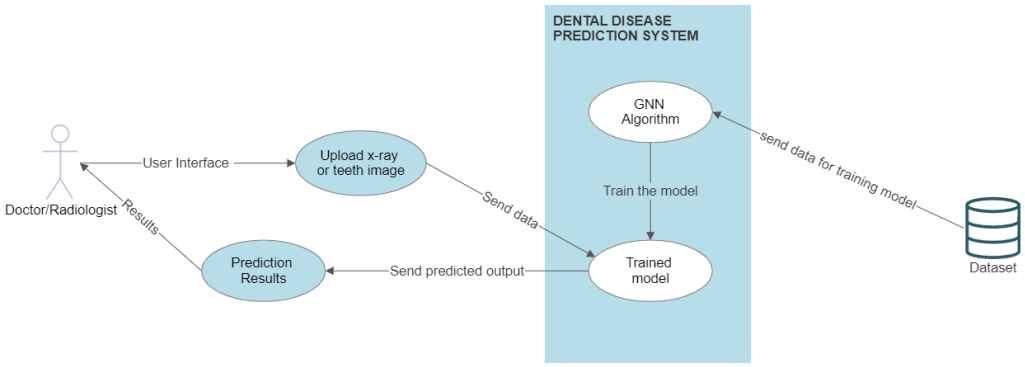


Figure 5.2: Usecase Daigram

5.2.2 Methodology

Flow Chart

First, the Dataset has to be searched for, and the suitable datasets can be short listed. The Dataset used in this Work is Panoramic X - Ray images of Teeth Set of different patients with the associated classes. 2 or more datasets can be merged together for having more diverse data, therefore increasing the training of the model in turn increasing the accuracy of the model. After the data has been selected, it cannot be directly fed into the model, it needs to be converted into numeric form. Therefore, we should perform the pre-processing of the dataset. Therefore, an image will be converted into an array. After the pre-processing, the data should be split into the training and the testing part in order to find the accuracy of the model. Therefore, the training and testing split takes place, after a shuffling of the dataset has taken place.

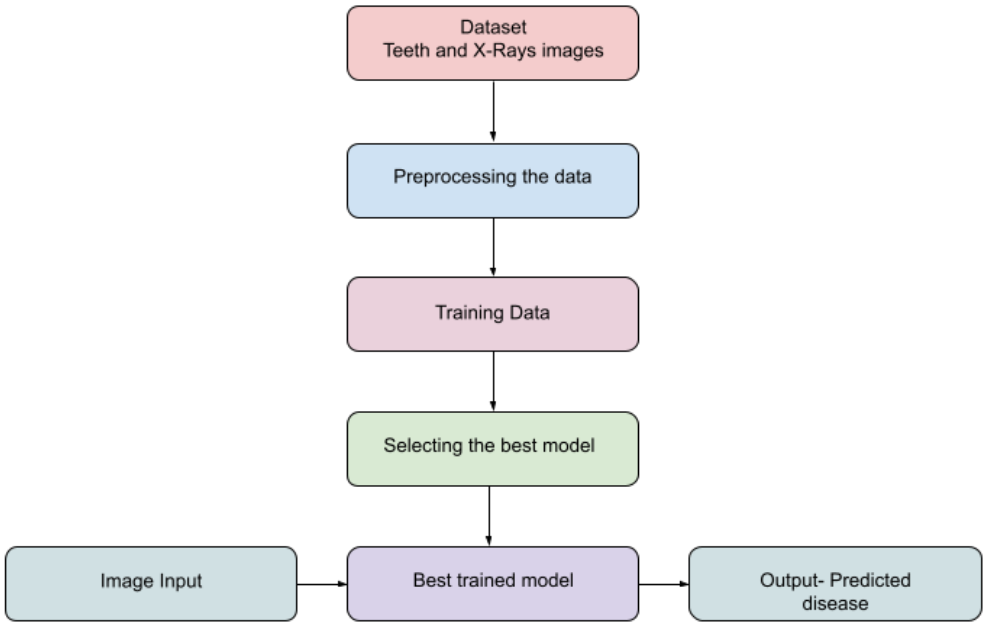


Figure 5.3: Flow Chart

In our Work, we have many algorithms such as GNN, CNN etc. Therefore, we execute the data in each of the models and record the accuracy metrics. The model which gives the highest accuracy is chosen as the Production Algorithm and is used for the prediction of Dental Diseases.

5.3 Dental Disease Prediction with Python

5.3.1 Data Collection

The dataset used consists of 116 patient's panoramic dental X-rays that were obtained from the Noor Medical Imaging Center in Qom, Iran. The images are de-identified and anonymous to protect the patients' privacy. These X-rays cover a wide range of dental disorders, including healthy teeth as well as partially and completely edentulous instances.

The dataset contains a total of 2057 X-ray images, each of which has been segmented manually by two dentists to highlight the mandibles in each patient. The dataset includes X-ray images of patients with various dental diseases, such as Caries, Gingivitis, Tooth-Discoloration, and Dental Ulcer. There are also X-ray images of patients who do not have any dental disease, referred to as "normal" in the dataset. Additionally, the dataset includes X-ray images of patients with general diseases.

The distribution of the dataset is shown in Figure 1, which illustrates the number of X-ray images for each type of disease, as well as for normal and general diseases. The dataset has 219 images of patients with Caries, 270 images of patients with Gingivitis, 183 images of patients with Tooth-Discoloration, and 265 images of patients with Dental Ulcer. There are 666 X-ray images of patients with normal teeth and 454 X-ray images of patients with general diseases.

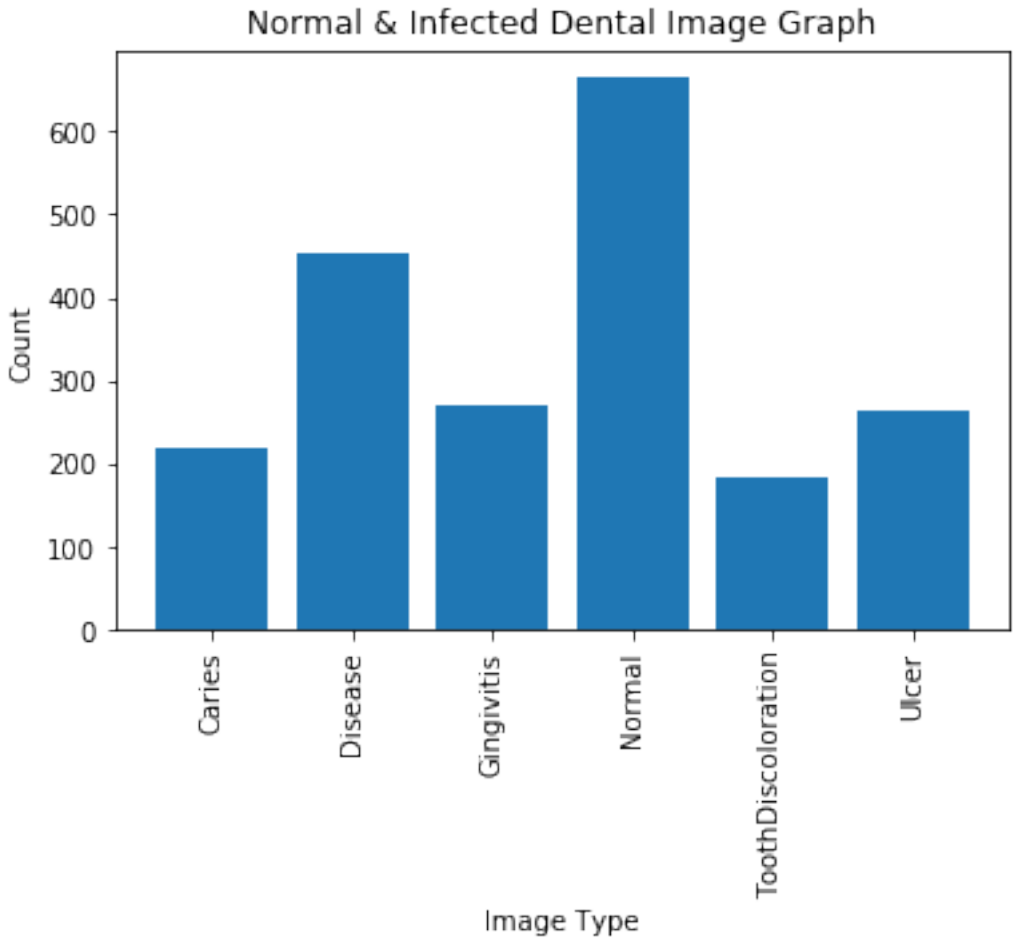


Figure 5.4: Distribution of Dental Diseases in the Chosen Dataset

This dataset can be useful for developing machine learning models to aid in the diagnosis and classification of dental diseases. By training on this dataset, it may be possible to create models that can accurately predict the presence of different dental diseases from X-ray images, which could be used to improve dental care and treatment planning.

5.3.2 Pre-processing

The User can either upload his own Model, or can iterate through all the images present in the given dataset, and form a custom model. An option of pickling and saving the model is also provided for easier access for the User.

In the Pre-Processing Phase, the model iterates through the given images in the Dataset. Using the $cv2$ library, the image is resized, and then converted into a numpy array. The numpy array is subjected to resizing by using the Red-Green-Blue (RGB) color images. The label of the image is computed. The Pre-Processed image is pushed into an array X, which contains all the images. The labels of the images in X are stored in the corresponding location in array Y.

Data Visualization provides the User with an insight into the type of Data found in the Dataset. Therefore, the Data is visualized to segment the various classes and the total number of Images present in the Dataset. This process is also done to avoid a biased conclusion i.e. If there is a huge variation in the number of entities in comparison between data classes, there is a high possibility that the prediction will favor the class with majority values in the dataset.

Here, the image array, or the array X is converted from a pandas framework into a floating array type. The possible values for each pixel are in range of $0 - 256$. A color code is represented by each digit. The computation

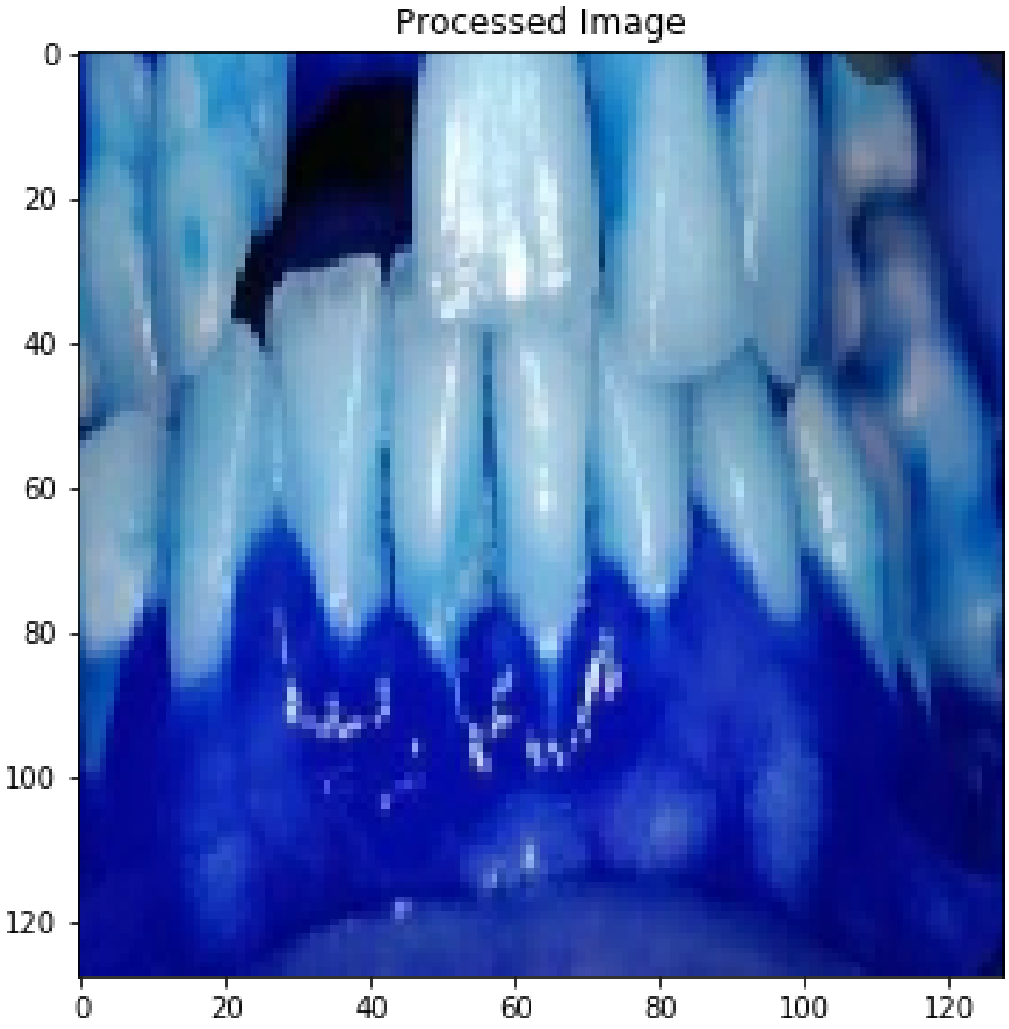


Figure 5.5: An example of the Pre-Processed Image

of high numeric values may become more difficult when using the image as-is and running it through a Deep Neural Network.

The data is normalized to fall between [0,1] to lessen this, the calculations will be simpler and quicker because the numbers will be tiny. Since pixel values vary in range of $0 - 256$, the range is 255 except for 0. Therefore, multiplying all of the values by 255 will change the range to be $0 - 1$.

After normalization, the Dataset is subjected to a shuffling process. This

process is done because as a result, the observations will be assigned to the training and test data at random. Now split the dataset into a training and validation phase. Here, the 80-20 split has been adopted i.e. 80% of the Data is used for training and 20% of the data is used for testing.

5.3.3 Algorithms

Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNNs) are a type of deep neural network that excels at image classification and recognition. The CNN method employs a layered architecture, with each layer composed of a series of convolutional filters that conduct feature extraction and mapping. The CNN algorithm is described in detail below:

- *Input* : The CNN algorithm is fed a 2D matrix of pixel values that represents an image as input.

- *Convolutional Layer* : The convolutional layer is the first layer of a CNN. This layer applies several kinds of filters to the input image in order to generate a set of feature maps. Each filter is a small weighted matrix that is convolved with the input image to create a new matrix known as a feature map. This layer's job is to find local patterns or characteristics in the input image, such as edges, corners, and blobs.The number of filters and their size are hyperparameters that can be changed to improve network performance.

- *Activation Function* : Following the convolutional layer, each feature map is subjected to an activation function. The Rectified Linear Unit (ReLU) function is the most commonly used activation function, which

uses the function f(x) = max(0, x) to each pixel value in the feature map. The activation function's objective is to bring nonlinearity into the network, allowing it to learn more complicated patterns and relationships in the input.

- *Pooling Layer* : The pooling layer is the next layer in a CNN. This layer decreases the size of feature maps by conducting a pooling operation on a small section of the feature map, such as max pooling or average pooling. This minimizes the network's computational complexity and aids in the prevention of overfitting.

- The convolutional, activation, and pooling layers should be repeated multiple times in order to extract progressively complex features from the input image.

- *Fully Connected Layer* : From the previous step, the output is vectorized and sent via one or more fully connected layers. These layers do classification by learning a nonlinear function that maps attributes of inputs to output classes. Each neuron in the fully connected layer is connected to all the neurons in the previous layer.

- *Output* : The CNN algorithm's ultimate output is a probability distribution over all possible classes derived by applying the softmax function to the output of the last completely connected layer.

Working of CNN:

Convolutional and max-pooling layers in the first section serve as the feature extractor. The fully connected layer, which does non-linear modifications of the retrieved features and serves as the classifier, makes up the second component.

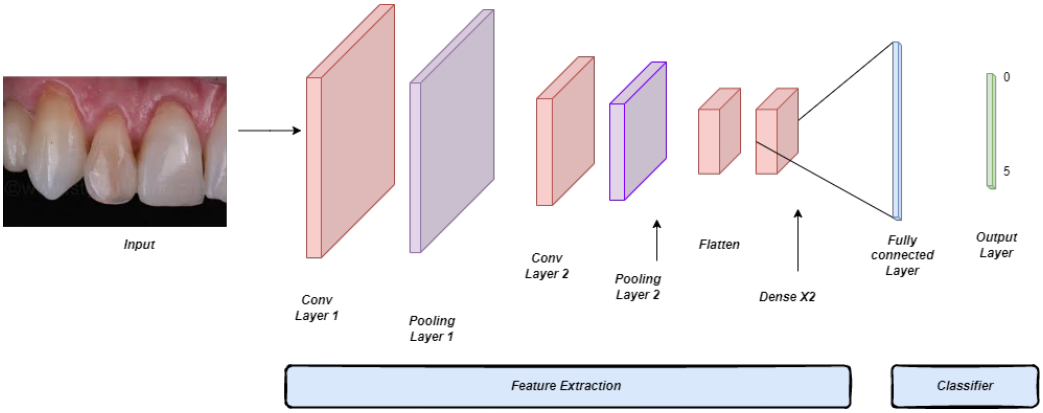


Figure 5.6: Architecture of CNN

The convolutional layer can be compared to CNN's eyes. The neurons in this layer scan for particular characteristics. They produce a high activation if they detect the features they are seeking for. In image processing, an image chunk of size $k \times k$ and center it at point (x, y) to calculate convolution at that location (x, y).

The convolution filter, which is also sized k x k, is used to multiply the values in this chunk one by one before adding them all together to produce a single output. Most frequently, the pooling layer is

applied right after the convolutional layer to lower the spatial size (only width and height, not depth).

First, the sequential object is defined. Using this sequential object, the image as a stack of layers is obtained. Exactly one input tensor and one output tensor. The Convolutional 2D is being used here. The Convolution 2D takes a feature and gets the feature map of the same. When doing the Convolutional 2D, the input function 'Relu' is used, where it is used to remove the negative values and replace them with zero value. It is also referred to as the activation function.

Convolutional 2D is followed by the Max Pooling Layer, so that more

insight can be obtained in the feature map. Max Pooling takes the max values from the sub matrices and forms a new matrix. The process is repeated to get accurate features. The next step is the 'Flatten' step. This step is done in order to convert the matrix into a single dimensional array i.e. 1 column matrix.

The Dense function is invoked. Each neuron in the simple layer of neurons known as the Dense Layer receives information from every cell in the layer below it. The output of the Dense model will be a classifier(0-5). Here, an option has been provided to pickle the model and save the model as a JSON file. This is done to save time, as the above steps can be avoided, and directly the model can be invoked and the prediction algorithm can be run.

CNN In Graphs So, we have seen Convolutional Neural Networks in the previous section, and how the different layers are associated with the Algorithm. Now, let us see how CNN is being applied to Graphs. A Graph can have many nodes, and the nodes are inter-connected by edges. Now, if an edge is present between 2 nodes, it relates that the 2 nodes are related to each other.

Applying CNNs to graphs also presents some challenges. For example, graphs can have different sizes and structures, which can make it difficult to design a fixed-size filter. Additionally, the lack of a natural ordering of nodes in a graph can make it challenging to apply convolutional operations. Therefore now we have to apply the CNN algorithm to the Graph.

For Applying the CNN Algorithm, first we have to convert the Graph into some numerical format, so that the algorithm can be applied. Therefore, the Adjacency Matrix of the Graph is created, which is a $N \times N$ matrix, where N denotes the number of nodes present in the Graph. In the $N \times N$ matrix, '$1'$ represents there is an edge between 2 nodes, '0' denotes that there is no

edge between the nodes.

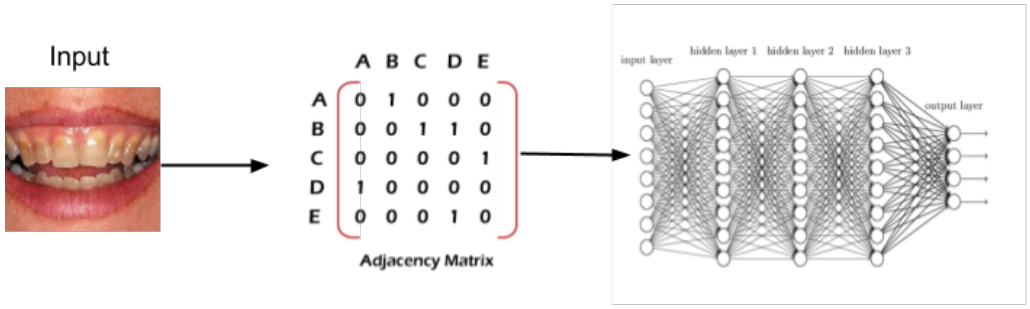


Figure 5.7: Working of CNN in Graphs

Each node is assigned a feature vector, and the convolution operation applies a filter to each node and its neighboring nodes. The resulting feature vectors for each node are then aggregated to produce a new feature vector for each node. This method is done for each node present in the system.

Graph Convolutional Networks(GCN)

GCNs (Graph Convolutional Networks) are a kind of neural network that works with graph-structured data. The basic idea behind GCNs is to execute convolutions over a graph structure by utilizing a supervised learning technique to learn the weights of the convolutional filters.

The GCN algorithm is described briefly below:

- *Input* : The GCN algorithm takes as input a graph represented by an adjacency matrix A and a feature matrix X, where A is a sparse binary matrix encoding the edges between nodes and X is a dense matrix encoding the features of each node.

- *Convolutional Operation* : To obtain features and learn node embeddings, the GCN algorithm performs a convolutional operation on the input graph. The operation takes into account the neighboring nodes and their features, and applies a filter to each node to produce a new feature. The convolutional operation is represented by:

$$h_i^{(l+1)} = \sigma(\sum_{j=1}^{n} \frac{1}{\sqrt{d_i d_j}} A_{ij} h_j^{(l)} W^{(l)}) \tag{5.1}$$

  where $h_i^{(l)}$ is the node embedding for node i at layer l, σ is an activation function such as ReLU, d_i is the degree of node i in the graph, and $W^{(l)}$ is the weight matrix at layer l.

- *Pooling Layer* : After several convolutional layers, This layer is performed to decrease the size of feature maps by conducting a pooling operation on a small section of the feature map, such as max pooling or average pooling. This minimizes the network's computational complexity and aids in the prevention of overfitting.

- The convolutional and pooling layers should be repeated multiple times in order to extract progressively complex features from the input image.

- *Fully Connected Layer* : From the previous step, the output is vectorized and sent via one or more fully connected layers. These layers do classification by learning a nonlinear function that maps attributes of inputs to output classes. Each neuron in the fully connected layer is connected to all the neurons in the previous layer.

- *Output* : The GCN algorithm's ultimate output is a probability distribution over all possible classes derived by applying the softmax function to the output of the last completely connected layer.

Graph Neural Network (GNN)

In the past ten years, neural networks have experienced tremendous success. Although many data sets in the real world contain underlying graph structures that are not Euclidean, early Neural Network variations could only be built using regular or Euclidean data. New developments in Graph Neural Networks have been made possible by the non-regularity of data structures. Graph Neural Networks are one of the many variations of Graph Neural Networks that have been developed over the last several years.

Graph Neural Networks (GNNs) are neural networks that are designed to operate on graph-structured data. The algorithm is divided into the following steps:

- *Input*: A graph with an adjacency matrix A and feature matrix X. A represents the edge connections between nodes, while X represents the feature vectors associated with each node.
- *Initialization*: The feature matrix X is first transformed using a weight matrix W. This transformation enables the network to learn new features based on the existing feature vectors.
- *Message Passing*: The message passing step involves passing information between neighboring nodes in the graph. At each iteration, each node aggregates information from its neighbors, and then updates its own feature vector based on the aggregated information. This process can be repeated multiple times to allow for more complex interactions between nodes.
- *Aggregation*: The aggregation step involves combining the updated feature vectors from each node in the graph to produce a single output.

- *Output*: The final output of the network can be used for tasks such as node classification, link prediction, or graph-level classification.

The GNN algorithm can be trained using backpropagation and gradient descent, similar to other neural networks. By training on labeled data, the network can learn to make accurate predictions on new, unseen data. GNNs can use a variety of techniques for message passing and aggregation, such as attention mechanisms, graph pooling, and edge convolution. This allows them to be adapted to a wide range of tasks and datasets.

Overall, the GNN algorithm enables neural networks to operate on graph-structured data and learn complex interactions between nodes. This makes it a powerful tool for tasks such as node classification, link prediction, and graph-level classification.

Working of GNN:

Consider a Graph in which the nodes have an edge between each other. The edge represents similarity between the 2 nodes. Each node represents a feature vector that represents the input data. These feature vectors are initialized based on some known information about the data. Next to get an accurate feature vector Message passing scheme is required. In order to perform forward propagation or Message passing in this computational graph, 3 steps have to be performed:

1. Initialize the activation unit: During message passing, the feature vector of a node(Xv) gets updated by combining the previous feature vector with the aggregate of the feature vectors of the neighboring nodes and it is represented as h_v^0

$$h_v^0 = X_v(Featurevector) \tag{5.2}$$

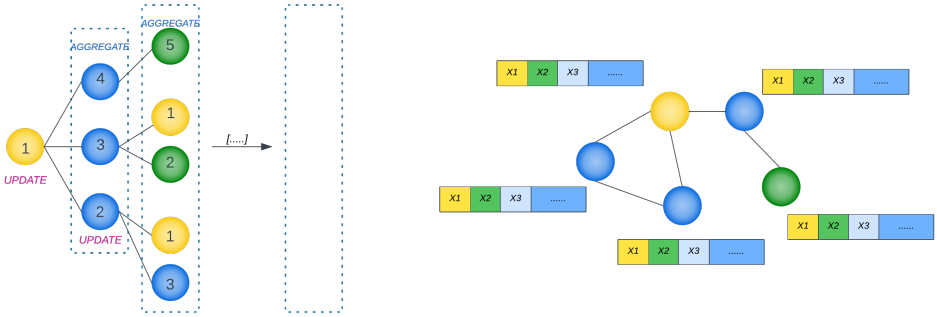


Figure 5.8: Working of GNN

2. This message passing is repeated several times until the nodes coverage to a stable state h_v^k.

$$h_v^k = \sigma(W_k \Sigma \frac{h_u^{k-1}}{N(v)} + B_{kh_v^{k-1}}) \text{where k = 1,...,k-1} \tag{5.3}$$

σ : the non-linearity activation that is performed on the two parts.

There are two parts for this equation:

The first part is basically averaging all the neighbors of node v.

$$W_k \Sigma \frac{h_u^{k-1}}{N(v)} \tag{5.4}$$

W_k : Weight matrix associated with the connections between the previous layer (u) and the current layer (v).

h_u^{k-1}: Represents the output of the previous layer, which is used as input to the current layer.

$N(v)$: Represents the node v.

The second part is the previous layer embedding of node v multiplied with a bias Bk, which is a trainable weight matrix and it's basically a self-loop activation for node v.

$$B_{kh_v^{k-1}} \tag{5.5}$$

B_k: This is a bias term associated with the current layer. It's indexed by "k" to indicate that there may be different bias terms for different layers.

3. Final layer: The final layer thus formed after layering is represented as Z_v.

$$Z_v = h_v^k \tag{5.6}$$

This process is called layering in GNN. After completing layering, the output graph will be formed as shown in 5.8. Now, Once the node features get updated the GNN model can be used for prediction based on the final feature vectors for each node in the graph. This can be implemented using Keras and Tensorflow.

Chapter 6

Web Application

Python web frameworks are software tools that are designed to make it easier to develop web applications in Python. These frameworks provide a set of tools and conventions that help developers to build web applications quickly and efficiently.

There are many different Python web frameworks available, each with its own strengths and weaknesses. Some of the most popular Python web frameworks include Flask, Django, Pyramid, Bottle, and CherryPy.

This Application is designed to help users identify whether a image of teeth contains disease or not. It uses a machine learning and deep learning model trained on a large dataset of teeth images to predict the given image as either normal or not. The model uses graphical image processing techniques to analyze each node in the given input image and feeds it to GNN model for prediction of dental disease.

In our application, we have utilized the Flask web framework. Flask is a Python-based web framework that is lightweight and provides flexibility in creating web applications. It has been designed to handle HTTP requests and responses, making it a popular choice for developing server-side applications

that can be accessed through a web browser or mobile app.

Flask has a minimalistic set of features, which makes it easy to learn and use. It offers support for routing, templating, and database integration, which are essential components in building web applications. Flask also has a built-in development server that makes it easy to test and debug applications locally.

One of the strengths of Flask is its flexibility, as it can be used to build a wide range of applications from small personal projects to large-scale web applications. It also supports a variety of deployment options, including deployment to cloud platforms such as Heroku and AWS.

In our application, we have leveraged the strengths of Flask to develop a web-based dashboard that displays data from various sources. Flask has provided us with the necessary tools and conventions to build the application quickly and efficiently. We have utilized Flask's support for database integration to store and retrieve data, as well as its routing capabilities to handle HTTP requests and responses.

Overall, Flask has been an essential component in the development of our application, and its flexibility and simplicity have made it a popular choice for many web developers.

6.1 Frame work

The implementation of a web application that uses Flask web framework to perform image classification. The application receives an image from the user, processes it using a pre-trained graph convolutional neural network model, and returns the predicted label and score for the image.

To begin with, the necessary Python modules are imported at the start

of the code, including Flask, OpenCV (cv2), NumPy, and Keras. The Flask web application object is created with the name "app", and the location of the HTML template files is specified as "templates". The secret key for the Flask application is also defined.

The main route of the Flask application is defined using the "@app.route" decorator, with the path set to "/" indicating the root directory of the application. The route handles both GET and POST requests, with the POST request used for uploading the image file and performing image classification.

The image file is uploaded using the "request.files" attribute and the filename is obtained using "secure_filename". The uploaded image is then read using OpenCV's "imread" function, resized to a shape of (32,32), and converted to a NumPy array. The array is then reshaped to a shape of (1,32,32,3) to match the expected input shape of the pre-trained GNN model.

The pre-trained GNN model is loaded using Keras' "load_ model" function, and the processed image is passed through the model using the "predict" method. The output is a probability distribution over the possible labels for the image, which is converted to a predicted label using NumPy's "argmax" function.

The predicted label, along with its corresponding score and the URL for the uploaded image, is passed to the "result.html" template using Flask's "render_template" function. The "result.html" template displays the predicted label, score, and uploaded image.

In the event of a GET request, the "index.html" template is displayed, which allows the user to upload an image file for classification.

Finally, the Flask application is run using the "app.run" method, with debug mode set to "False".

Overall, the application provides a simple and effective way to build an

image classification web application using Flask and pre-trained deep learning models.

6.1.1 Index Page

The index page comprises six primary sections, namely the header section, home section, about section, services section, more information section, and predict section.This page takes the image as input.

Header Section

The header section contains the logo of the website and a navigation menu that enables users to navigate to different sections of the website. The header also has a link button that directs users to the Predict section of the website. The header is fixed at the top of the page, which means that it remains visible as the user scrolls down the webpage.

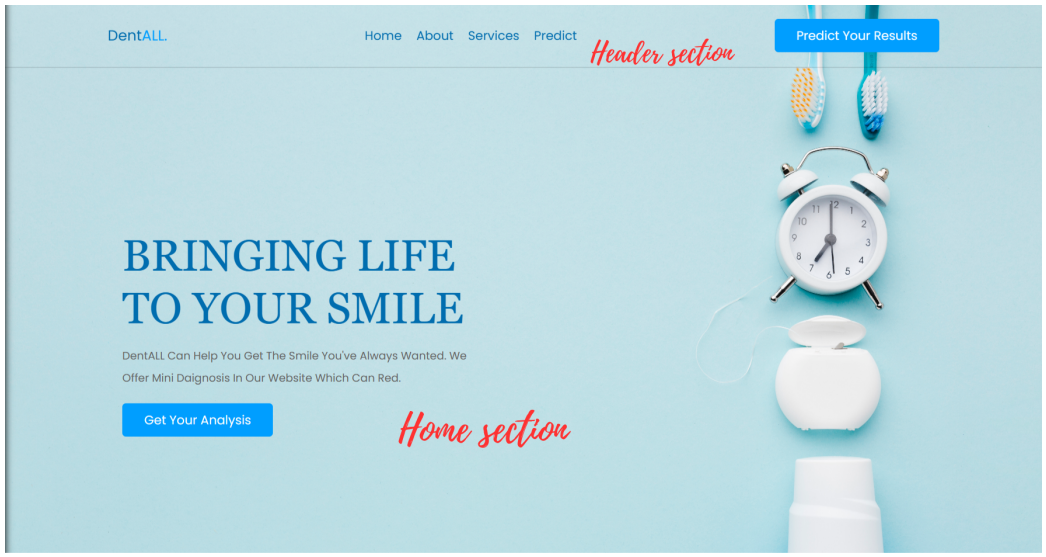


Figure 6.1: Header and Home section

Home Section

The home section is the first section of the webpage and it contains a hero image and a call-to-action button. The hero image contains a background image and a text overlay that highlights the key value proposition of the website, which is to bring life to your smile. The call-to-action button links to the Predict section of the website where users can get a personalized analysis of their dental health.

About Section

The about section provides information about the website and its services. It contains an image of a dentist and a brief description of the website's mission to help users achieve a quintessentially oriented smile. The section also contains a call-to-action button that links to the Predict section of the website.

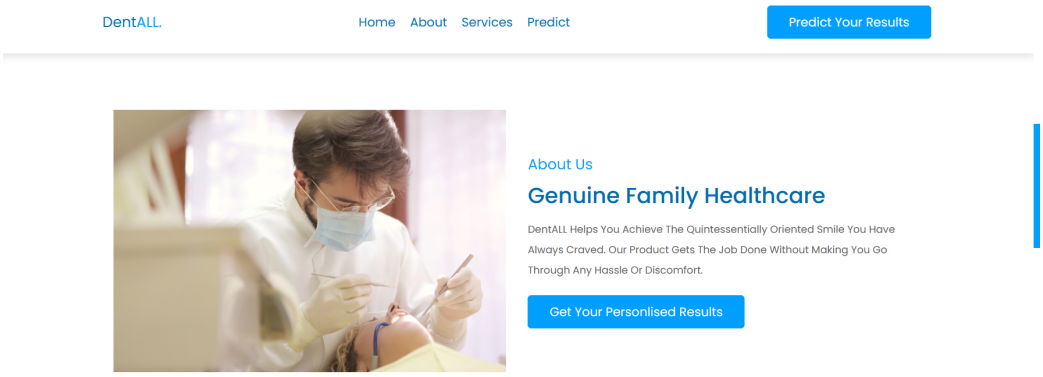


Figure 6.2: About section

Services Section

The services section provides information about the different dental problems that the website can diagnose. It contains six boxes that represent different

dental problems such as tooth discoloration, caries, gingivitis, ulcer, disease, and healthy teeth. Each box contains an image and a title that describes the dental problem.

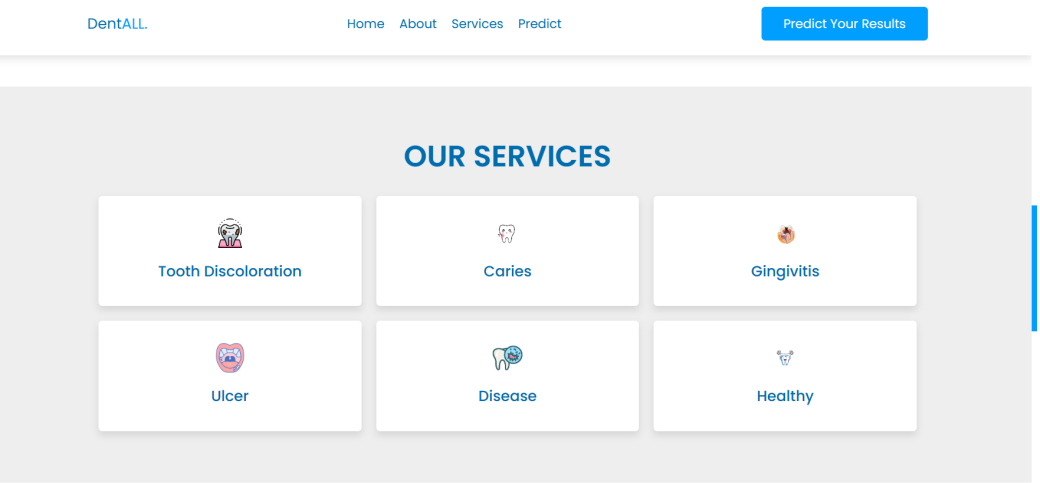


Figure 6.3: Service section

More Information Section

The more information section provides detailed information about the dental problems that the website can diagnose. The section contains two boxes that describe tooth discoloration and caries. Each box contains a heading that describes the dental problem and a paragraph that provides information about the causes and symptoms of the dental problem.

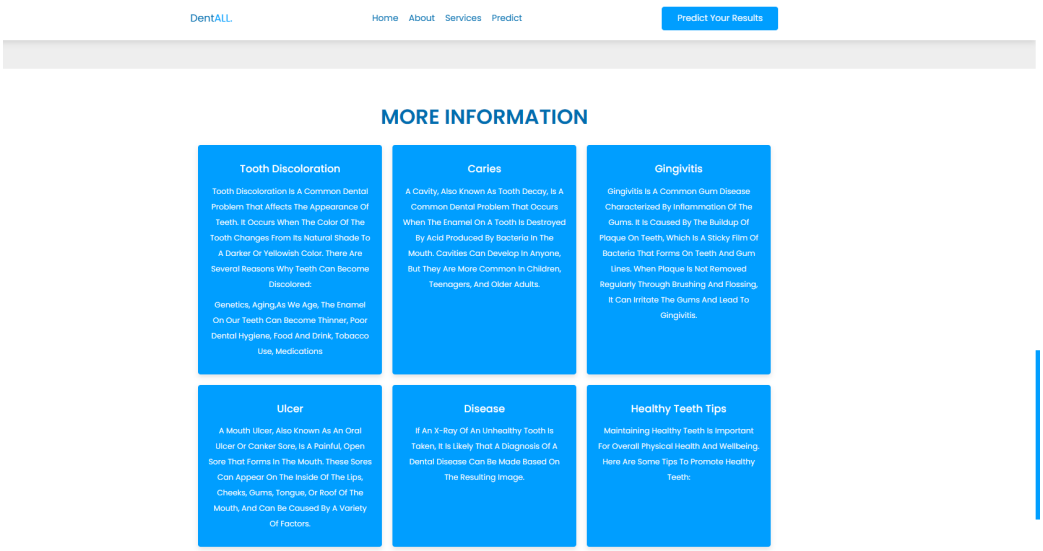


Figure 6.4: More Information section

Predict Section

The predict section provides users with a personalized analysis of their dental health. The section contains a form that users can fill out to get their analysis. The form asks users to provide information such as their age, gender, and symptoms. Once users submit the form, the website provides them with a diagnosis of their dental health.

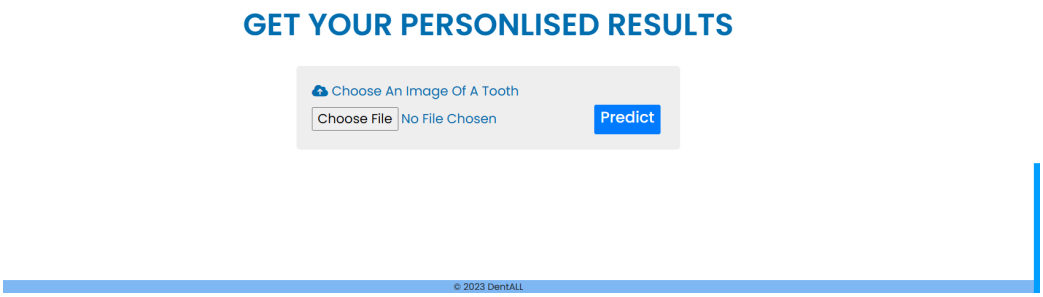


Figure 6.5: Predict section along with footer section

Overall, the webpage is designed to provide users with information about their dental health and to help them diagnose common dental problems. The webpage has a clean and modern design that makes it easy for users to navigate and find the information they need.

6.1.2 Result Page

The Result page that displays the results of a dental diagnosis. The code includes several sections mainly header section, home section and information section.Header section is same as in the index page with only result and information tabs.

Home Section

The home section displays the result of the diagnosis in a text format, which includes the diagnosis label and score. The section also displays an image related to the diagnosis.

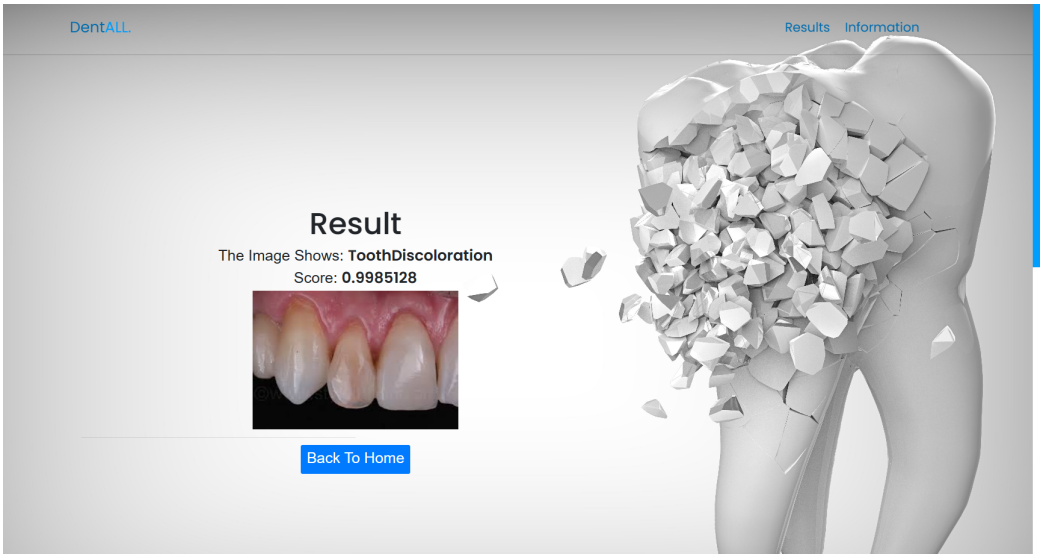


Figure 6.6: Home Section

Information Section

The information section presents additional details about the diagnosis, which includes the label of the diagnosis and its associated content.

Lastly, the footer section of both pages displays the copyright information of the webpage. The code also includes links to external CSS files for styling the page, and JavaScript files for adding interactivity and retrieving data.

Figure 6.7: Information Section

6.2 Functionality

This web application using Flask, which allows users to upload an image, and the application uses a pre-trained model to predict the class label of the image. The predicted label, confidence score, and the uploaded image are then displayed on a result page.

The uploaded image is first saved in a directory named 'static/uploads'. If this directory does not exist, the code creates it. The image is then read using OpenCV and resized to 32x32 pixels. The pixel values are then normalized by dividing them by 255, as is common practice with image data. The image is then passed through a pre-trained Graph Convolutional Neural Network (GNN) model to predict the class label of the image.

The predicted label and the confidence score are then displayed on the result page, along with the uploaded image, which is accessed using a URL generated using the url_for() function. The HTML templates for the index and result pages are stored in the templates folder, which is specified using the template_folder parameter when creating the Flask app object.

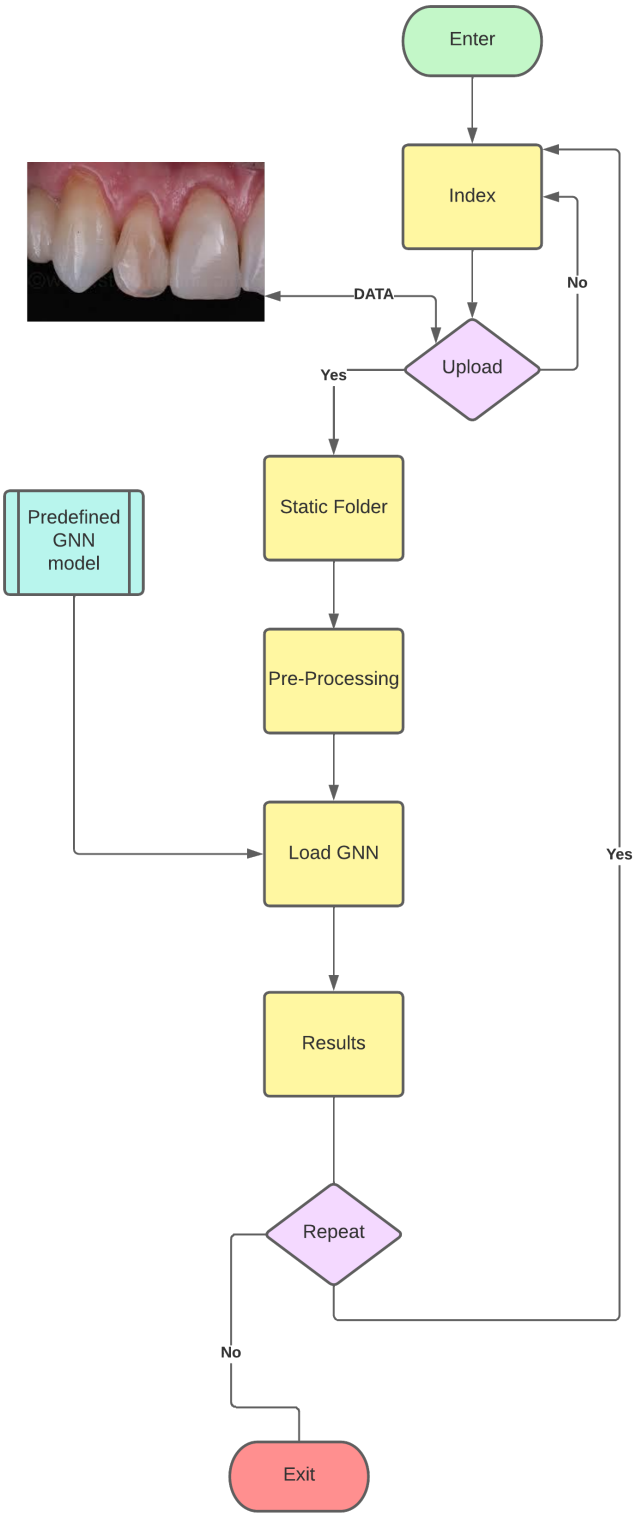


Figure 6.8: Flow chart of Web Application

Overall, the 6.8 flowchart demonstrates the functionality of web application that allows users to upload images and perform image classification using a pre-trained deep learning model.

Chapter 7

Result and Analysis

Descriptive statistics

For the experiment, the entire dataset was used, with the Train and Test Data split at 70% and 30%, respectively.

Based on these statistics, it can be seen that the GNN model outperforms the CNN model across all four metrics. The GNN has higher accuracy, precision, F1-score, and recall, indicating better overall performance compared to the CNN model.

CNN	
Accuracy	93.689320
Precision	94.259378
F1-Score	94.032136
Recall	93.840778

Table 7.1: Descriptive statistics for CNN

GNN	
Accuracy	96.029126
Precision	97.499451
F1-Score	97.569626
Recall	97.679837

Table 7.2: Descriptive statistics for GNN

Model evaluation metrics

A ***confusion matrix*** can be used to compare the performance of both the models. A method for summarising the performance of a classification algorithm is the confusion matrix. Classification accuracy alone can be deceptive if the dataset has more than two classes or unequal numbers of observations in each class. The strength and weakness of the model can be determined using confusion matrices.

In this experiment, a confusion matrix of size 6x6 is obtained. The number of classes in the dataset determines the size of the confusion matrix.

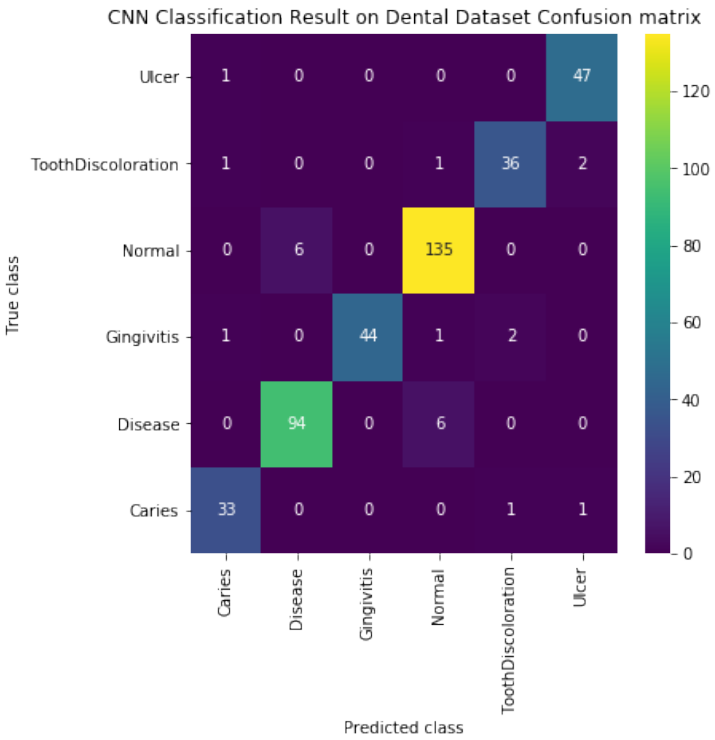


Figure 7.1: Confusion Matrix for CNN

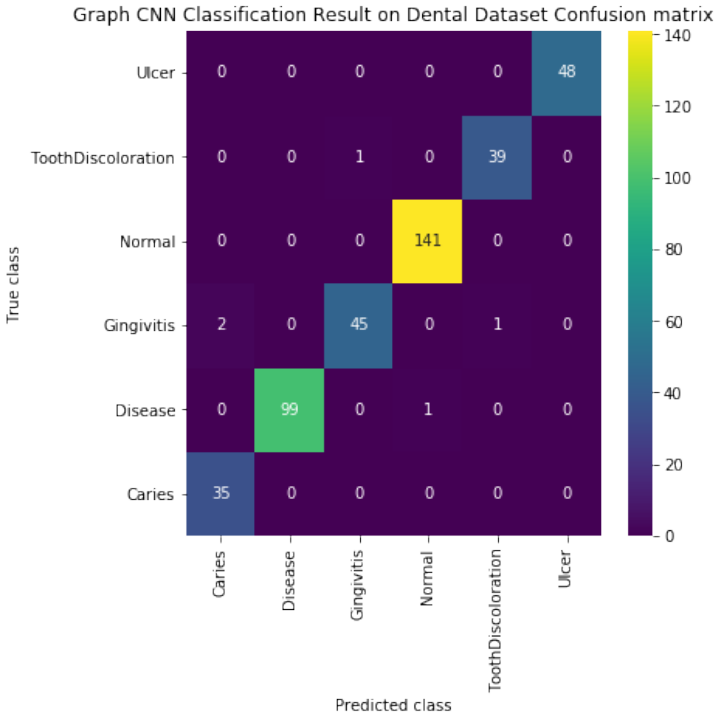


Figure 7.2: Confusion Matrix for GNN

Model comparison

These results suggest that the GNN model is more effective than the CNN model for the given task. However, it is important to note that these results are based on the specific dataset and evaluation metrics used in the study. Other datasets or metrics may yield different results, and it is important to carefully consider the specific requirements and constraints of the problem at hand when selecting a model.

In summary, based on the results presented in the tables, the GNN model is the preferred choice over the CNN model for this particular task. However, additional evaluation and testing may be necessary to fully validate the effectiveness and generalizability of the model.

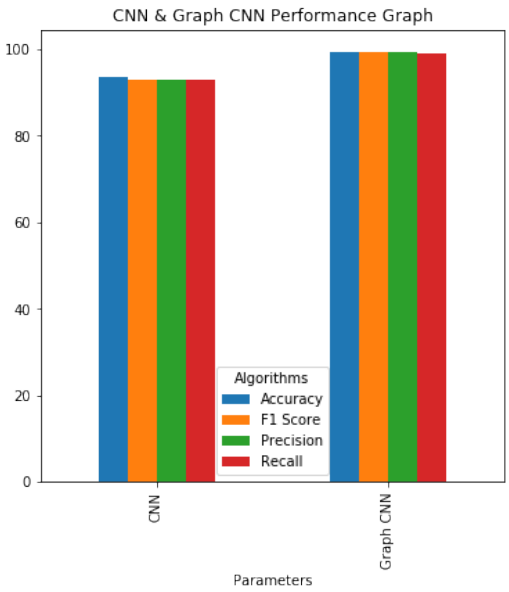


Figure 7.3: Bar Graph comparision betweeb CNN and GNN

Feature importance

In dental image classification, feature importance plays a critical role in improving the accuracy and interpretability of Graph Neural Networks (GNNs). The goal of dental image classification is to accurately diagnose and classify various dental conditions such as caries, periodontitis, and gingivitis. GNNs are a powerful tool for this task as they can effectively capture the complex relationships between the various dental features in the image.

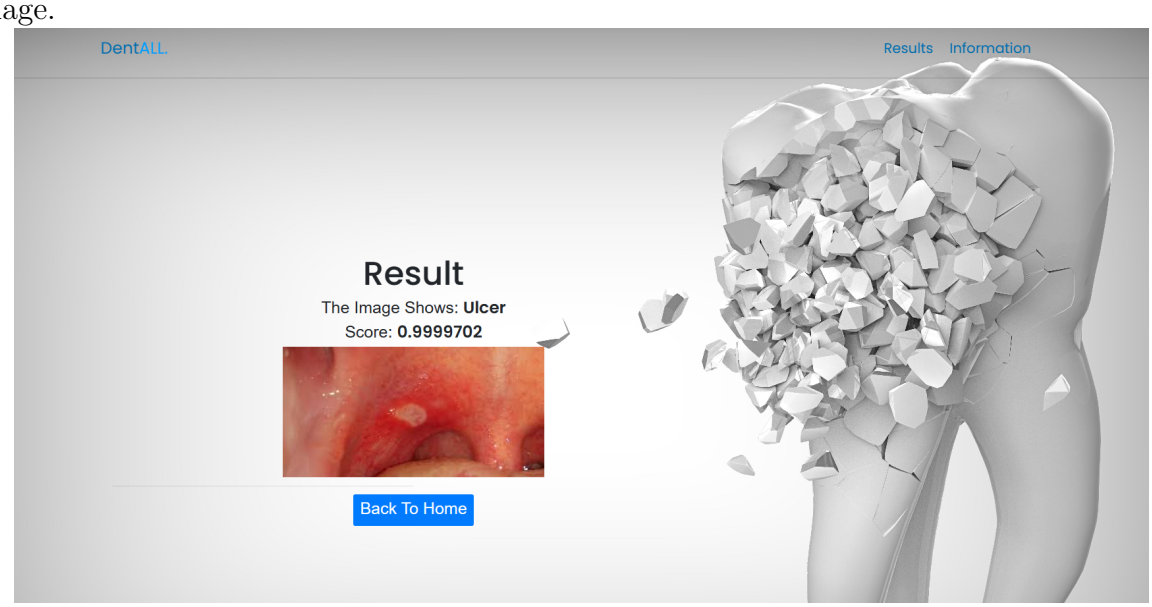
Feature importance in dental image classification refers to the relevance of each feature in the image towards predicting the dental condition. This can include various features such as the color of the tooth, the shape and size of the tooth, and the texture of the tooth surface. By identifying and ranking the most important features, we can better understand the underlying structure of the dental image and develop more accurate and interpretable GNN models.

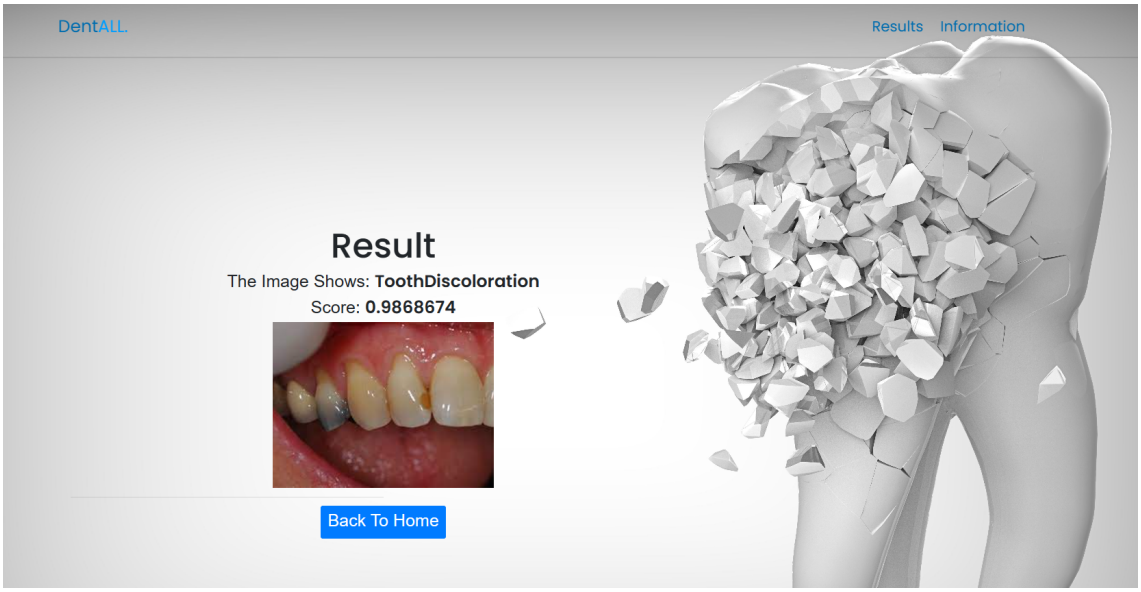
In this experiment tooth color have the highest ranking in feature importance.

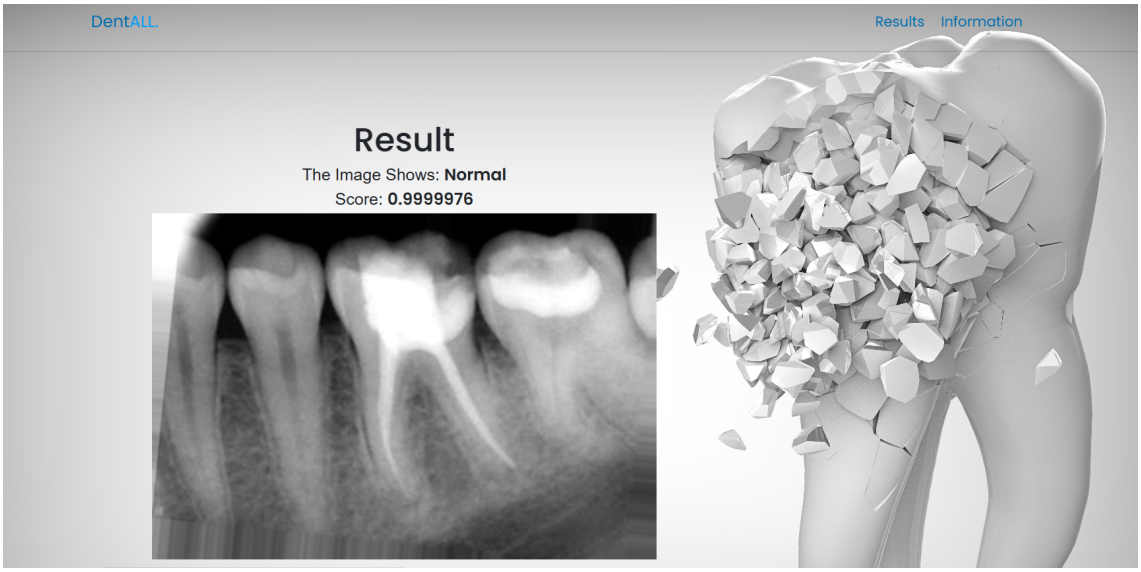
In addition to improving the accuracy of dental image classification, feature importance can also enhance the interpretability of GNN models. By visualizing the most important features, we can better understand the relationship between the dental features and the dental condition being classified. This can be particularly useful for dental professionals who need to make accurate diagnoses and develop appropriate treatment plans.

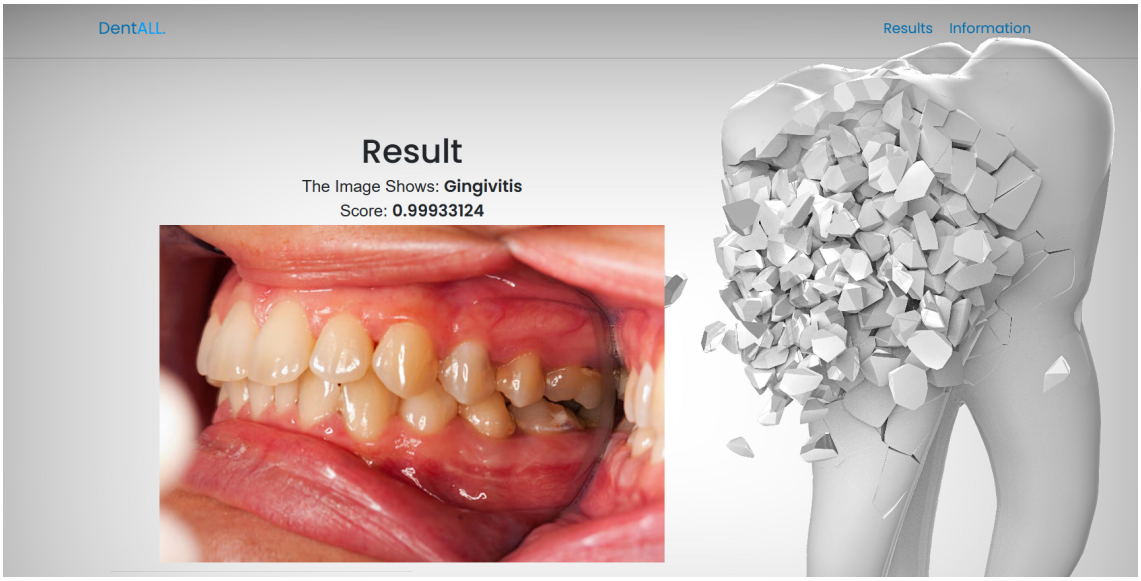
Visualization

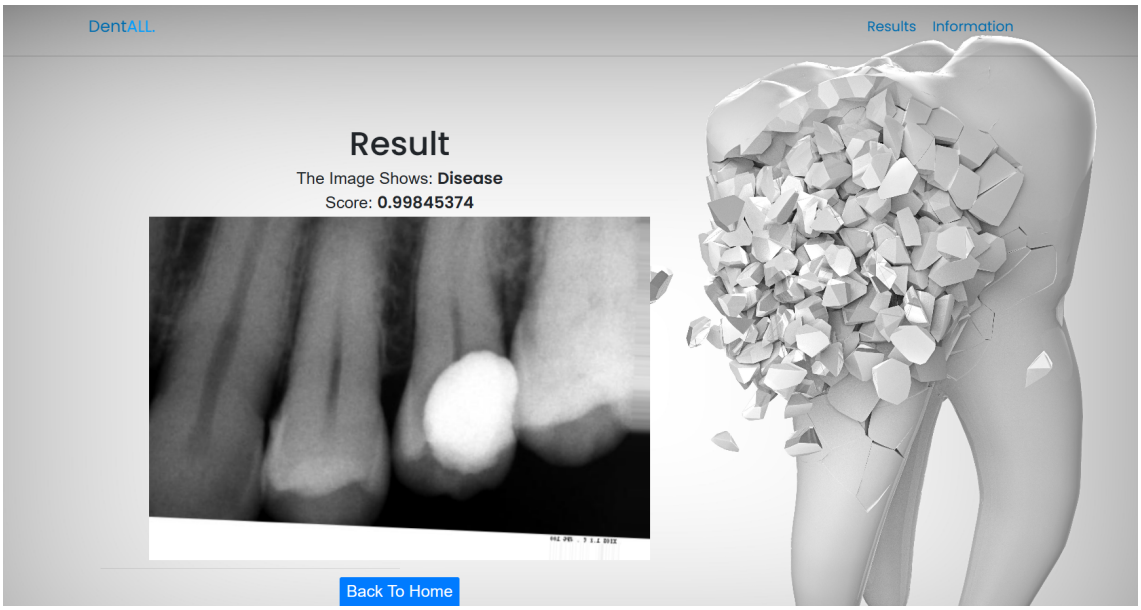
These below images shows the predicted disease along with score of that image.




DentALL.
Results
Information
Result
The Image Shows: ToothDiscoloration
Score: 0.9868674
Back To Home


DentALL.
Results
Information
Result
The Image Shows: Normal
Score: 0.9999976


DentALL.
Results
Information
Result
The Image Shows: Gingivitis
Score: 0.99933124


DentALL.
Results
Information
Result
The Image Shows: Disease
Score: 0.99845374
Back To Home

Chapter 8

Conclusion

The experimental results show the superiority of Graph-based approach over Normal CNN ones with respect to the prediction performance across various graphs. In contrast, compared to normal CNN approaches, these Graph-based approaches are less suitable when the graphs need fast processing. The computational time of GNN-based approaches is further affected when applied to large graphs. One perspective of this Project is to realize a good trade-off between prediction accuracy and computational time by developing a GNN-based link approach in a distributed and parallel environment. In the chosen dataset, there are only very few records of data. For developing an efficient model which can predict Dental Disease correctly, more data is required for training the model. Since there are only limited datasets available on the web, more data can be collected from Dental Hospitals and Laboratory. A serious point to be looked at is that the high accuracy given by the Deep Learning models can be due to the limited amount of data used to train the model. Since both the training and testing of the model is done by a 70/30 split, and both the data is from the same dataset, there is a chance of the result to be biased. Therefore, the model can be overfit to

that particular dataset, thereby the accuracy is really high in this study, and the accuracy by testing with images of other sources may be giving a lower accuracy and precision scores. By analysing the study, although GNN takes more time than CNN on larger datasets, keeping in mind the severity of the problem statement, the end user expects a more accurate result rather than a faster result. AN inaccurate result can lead to drastic circumstances, such as a healthy person receiving unwanted treatment, or the case of wrong disease prediction, where the wrong treatment for the present disease. Although this mechanism is proposed only as a preliminary analysis, still the prediction of wrong result can be dangerous. Therefore it can be concluded that in the prediction of Dental Diseases, the GNN model is more preferred than the CNN model. The approach can also be used to predict in heterogeneous graphs such as knowledge graphs.

References

[1] Kipf, Thomas Welling, Max. (2016). Semi-Supervised Classification with Graph Convolutional Networks.

[2] Hu, W., Catasta, M., Leskovec, J. (2020). Open Graph Benchmark: Datasets for Machine Learning on Graphs. arXiv preprint arXiv:2005.00687.

[3] Hamilton, William Ying, Rex Leskovec, Jure. (2017). Inductive Representation Learning on Large Graphs.

[4] Mircea Paul Muresan, Andrei Rzvan Barbura, "Teeth Detection and Dental Problem Classification in Panoramic X-Ray Images using Deep Learning and Image Processing Techniques" 2020 DOI: 10.1109/ICCP51029.2020.9266244

[5] Robson D. Montenegro, AdrianoL.I. Oliveira, George G. Cabral,Cintia R. T. Katz, Aronita Rosenblatt, "A Comparative Study of Machine Learning Techniques for Caries Prediction,", DOI: 10.1109/ICTAI.2008.138

[6] Dhruv Verma, Dr Sunaina Puri , Dr Srikanth Prabhu and Dr Komal Smriti , "Anomaly detection in panoramic dental x-rays using a hybrid Deep Learning and Machine Learning approach

[7] Lilian Toledo Reyes, Jessica Klöckner Knorst, Fernanda Ruffo, Thiago Machado, "Scope and challenges of machine learning-based diagnosis and prognosis in clinical dentistry: A literature review", DOI:10.18053/Jctres/07.202104.012

[8] Abhinav Duvvuri, Saran Sai C, Dheerai M, Simi Surendran, "Ensemble Based Predictive Model for Streaming Data"

[9] Jain A. K., Chen H., "Matching of dental X-ray images for human identification," Pattern Recognit. 37(7), 1519–1532 (2004). 10.1016/ j.patcog.2003. 12.016

[10] Lin P. L., Lai Y. H., Huang P. W., "An effective classification and numbering system for dental bitewing radiographs using teeth region and contour information," Pattern Recognit. 43(4), 1380–1392 (2010). 10.1016/j.patcog.2009.10.005

[11] Lin P. L., Lai Y. H., Huang P. W., "Dental biometrics: human identification based on teeth and dental works in bitewing radiographs," Pattern Recognit. 45(3), 934–946 (2012). 10.1016/ j.patcog. 2011.08.027

[12] S Abhishek, Harsha Sathish, Arvind Kumar, Anjali T, "Aiding the Visually Impaired using Artificial Intelligence and Speech Recognition Technology".

[13] Mahima Chowdary Mannava, Bhavana Tadigadapa, Devitha Anil, Aparna S Dev, Anjali T, "CNN Comparative Analysis for Skin Care Classification"

[14] S Abhishek, Harsha Sathish, Arvind Kumar, Anjali T, "A Strategy for Detecting Malicious Spam Emails using various Classifiers"

[15] Tharun Thomas Ajith, Ajay Krishnan C V, Nandakishore J, M S Ananthu Subramanian, Siji Rani S, "Enhanced Movie Recommendation using Knowledge Graph and Particle Filtering"

[16] F. Baldassarre, H. Azizpour Explainability Techniques for Graph Convolutional Networks ICML Workshop on Learning and Reasoning with Graph-Structured Representations (2019).

[17] Gopa Vasanth, Nishanth S, PVS S Alavandar, Paturi Vamsi Krishna, P P Venkat Sai, Siji Rani S, "Flick - Indigenous short video application"

Appendix A

Source code

A.1 Overview of Dataset and Data Acquisition

The Dataset used in the Work contains a total of 6 classes : 4 particular disease classes, 1 general disease class, and a Normal Class. In each folder of this class, the X-Ray images of the particular class are present. Initially, for obtaining the data, and for obtaining the index of the class, the getID() function is being used.

```
path = 'Dataset'
labels = ['Caries', 'Disease', 'Gingivitis', 'Normal', 'ToothDiscoloration', 'Ulcer']
def getID(name):
    index = 0
    for i in range(len(labels)):
        if labels[i] == name:
            index = i
            break
    return index
```

A.2 Vector Preparation and Optimization for Model Training

In order for training the model, we need the Input Vector X and the output vector Y. An option has been provided here to save the vectors in the 1st run of the Notebook into the local folder, so that in future running of the Notebook, the program automatically takes the saved vectors, thereby reducing the execution time of the Notebook. If there is no local vectors available, each image in the Dataset is read one by one, and is then resized into 32x32, and then converted into a numeric numpy array. Then, the image features are uploaded to X, and the label obtained through the getID() is stored correspondingly in Y. The vectors are then saved to the local.

```
X = []
Y = []
if os.path.exists("model/X.txt.npy"):
    X = np.load('model/X.txt.npy') #load X and Y data
    Y = np.load('model/Y.txt.npy')
else:
    for root, dirs, directory in os.walk(path):#loop all images in dataset folder
        for j in range(len(directory)):
            name = os.path.basename(root)
            if 'Thumbs.db' not in directory[j]:
                img = cv2.imread(root+"/"+directory[j]) #read each image one by one
                img = cv2.resize(img, (32,32)) #resize image
                im2arr = np.array(img)
                im2arr = im2arr.reshape(32,32,3)#resize image as color images by using 3 colours such as RGB
                label = getID(name)#get the label of the image
                X.append(im2arr) #add image to array as X features
                Y.append(label) #add label to Y array
    X = np.asarray(X) #convert images to numpy array
    Y = np.asarray(Y)
    np.save('model/X.txt',X) #save X and Y values
    np.save('model/Y.txt',Y)
classes, count = np.unique(Y, return_counts = True) #find normal and disease images count
```

A.3 Image Processing

The images of the Dataset should be processed before they can be used to train the model. The image as such cannot be uploaded to the model, it should be changed to a numerical format. Therefore, first the image is converted to type float. Now, The images of the Dataset should be processed

before they can be used to train the model. The image as such cannot be uploaded to the model, it should be changed to a numerical format. Therefore, first the image is converted to type float.

```
#images preprocessing and features extraction
X = X.astype('float32')
X = X/255 #normalizing images
indices = np.arange(X.shape[0])
np.random.shuffle(indices) #shuffling the dataset images
X = X[indices]
Y = Y[indices]
Y = to_categorical(Y)

img = cv2.imread("Dataset/Gingivitis/(27).jpg")
img = img / 255
img = cv2.resize(img, (128, 128))
plt.imshow(img)
plt.title('Processed Image')
plt.show()
```

A.4 Data Splitting

Using the train_test_split(), the Dataset is split for training and testing. In our work, we have used 80 percent of Data for training and 20 percent of Data for testing.

```
#splitting dataset into train and test
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
print()
print("80% images used for training & 20% images used for testing")
print("Training Images Size : "+str(X_train.shape[0]))
print("Testing Images Size  : "+str(X_test.shape[0]))
print()
```

A.5 Model Summary

A.5.1 CNN

In the prediction of Dental Diseases, first the CNN algorithm is being used with help of keras library. A total of 32 filters are defined. Then, the Convolution Layer is done, followed by the Max Pooling Layer. After iteration of both these layers, the vector is made into one dimension using the Flatten function. After the Flatten, we have to Dense the output. An Option is given here to pickle the model. When the model is pickled, in the subsequent executions, where instead of training the model, the saved model can be used for the prediction. This process acts as a time saver.

```
from keras.callbacks import ModelCheckpoint
#train & load CNN model
cnn = Sequential()#define sequential object
#define CNN layer with 32 filters
cnn.add(Convolution2D(32, (3, 3), input_shape = (X_train.shape[1], X_train.shape[2], X_train.shape[3]), activation = '
relu'))
cnn.add(MaxPooling2D(pool_size = (2, 2)))
cnn.add(Convolution2D(32, (3, 3), activation = 'relu'))
cnn.add(MaxPooling2D(pool_size = (2, 2)))
cnn.add(Flatten())
cnn.add(Dense(units = 256, activation = 'relu'))
cnn.add(Dense(units = y_train.shape[1], activation = 'softmax'))
#compile the model
cnn.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])
#start training model
if os.path.exists("model/model_weights.hdf5") == False:
    model_check_point = ModelCheckpoint(filepath='model/model_weights.hdf5', verbose = 1, save_best_only = True)
    hist = cnn.fit(X_train, y_train, batch_size=16, epochs=25, validation_data = (X_test, y_test), callbacks=[model_ch
eck_point], verbose=1)
    f = open('model/history.pckl', 'wb')
    pickle.dump(hist.history, f)
    f.close()
else:
    cnn.load_weights("model/model_weights.hdf5")
print(cnn.summary())
```

A.5.2 GNN

The CNN model is trained using 32 filters, and then the Convolutional Layer, the Max Pooling Layer and the Activation functions are all applied to the same. The model is created as per the above discussions in Section 5.5.1. Now, the model is being used for prediction. The calculateMetrics() is used for displaying the accuracy, precision metrics and the confusion matrix of the

model.

```
#training Graph CNN Algorithm
X1 = np.reshape(X, (X.shape[0], (X.shape[1] * X.shape[2] * X.shape[3])))
X_train1, X_test1, y_train1, y_test1 = train_test_split(X, Y, test_size=0.2)
graph_conv_filters = np.eye(32)
graph_conv_filters = K.constant(graph_conv_filters)
graph_cnn = Sequential()
#defining graph cnn layer with 128 filters
graph_cnn.add(GraphCNN(128, 1, graph_conv_filters, input_shape=(X_train1.shape[1],), activation='elu', kernel_regulari
zer=l2(5e-4)))
#defining another layer with 64 layer
graph_cnn.add(GraphCNN(64, 1, graph_conv_filters, input_shape=(X_train1.shape[1],), activation='elu', kernel_regulariz
er=l2(5e-4)))
graph_cnn.add(Dense(units = 256, activation = 'relu'))
#add the output layer
graph_cnn.add(Dense(units = y_train1.shape[1], activation = 'softmax'))
#compile the model
graph_cnn.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])
#train the model
print(graph_cnn.summary())
if os.path.exists("model/graph_model_weights.hdf5") == False:
    hist = graph_cnn.fit(X_train1, y_train1, batch_size=32, epochs=25, shuffle=True, verbose=2, validation_data = (X_t
est1, y_test1))
    graph_cnn.save_weights('model/graph_model_weights.h5')
    f = open('model/graph_history.pckl', 'wb')
    pickle.dump(hist.history, f)
    f.close()
else:
    graph_cnn = load_model("model/graph_model_weights.hdf5", compile=False)
```

A.6 Confusion Matrix of the Model

The below code is used to calculate the accuracy, precision, recall score and F score of the created model. The confusion matrix is also being displayed here.

```
#Calculate accuracy and other metrics
#function to calculate various metrics such as accuracy, precision etc
def calculateMetrics(algorithm, predict, testY):
    p = precision_score(testY, predict,average='macro') * 100
    r = recall_score(testY, predict,average='macro') * 100
    f = f1_score(testY, predict,average='macro') * 100
    a = accuracy_score(testY,predict)*100
    print()
    print(algorithm+' Accuracy  : '+str(a))
    print(algorithm+' Precision   : '+str(p))
    print(algorithm+' Recall      : '+str(r))
    print(algorithm+' FMeasure    : '+str(f))
    accuracy.append(a)
    precision.append(p)
    recall.append(r)
    fscore.append(f)
    conf_matrix = confusion_matrix(testY, predict)
    plt.figure(figsize =(6, 6))
    ax = sns.heatmap(conf_matrix, xticklabels = labels, yticklabels = labels, annot = True, cmap="viridis" ,fmt ="g");
    ax.set_ylim([0,len(labels)])
    plt.title(algorithm+" Confusion matrix")
    plt.ylabel('True class')
    plt.xlabel('Predicted class')
    plt.show()
```

A.6.1 GNN

```
#perfrom prediction using Graph CNN and then calculate accuracy and precision
predict = graph_cnn.predict(X_test)
predict = np.argmax(predict, axis=1)
y_test = np.argmax(y_test, axis=1)
calculateMetrics("Graph CNN Classification Result on Dental Dataset", predict, y_test)
```

The above code is used for predicting Dental Diseases using the GNN algorithm and then displays the accuracy metrics and the confusion matrix of the generated model.Similarly we can find out confusion matrix for CNN model.

A.7 Performance Comparison

A Bar Graph with the Accuracy, Precision, F1 score and Recall score of the CNN and the GNN algorithms are displayed here. Using this graph, a person can easily identify the most accurate and precise algorithm. Here, the GNN algorithm performs better than the CNN Algorithm in all accuracy metrics.

```
#display comparison graph of CNN and Graph CNN
df = pd.DataFrame([['CNN','Precision',precision[0]],['CNN','Recall',recall[0]],['CNN','F1 Score',fscore[0]],['CNN','Ac
curacy',accuracy[0]],
                   ['Graph CNN','Precision',precision[1]],['Graph CNN','Recall',recall[1]],['Graph CNN','F1 Score',fsc
ore[1]],['Graph CNN','Accuracy',accuracy[1]],
                  ],columns=['Parameters','Algorithms','Value'])
df.pivot("Parameters", "Algorithms", "Value").plot(kind='bar')
plt.title("CNN & Graph CNN Performance Graph")
plt.show()
```

A.8 Web Application

The User-Interface which will be used by the Dentist is done using flask. The GNN algorithm, which is having maximum accuracy and precision among all the other algorithms is loaded here. The testing image is being accepted from the Dentist, and is then pre-processed using the cv2 library. The image is then tested in the model, and the related class of the image is returned to the Dentist.

```python
from flask import Flask, render_template, request,url_for
import cv2
import numpy as np
from tensorflow.keras.models import load_model
import os
from werkzeug.utils import secure_filename

app = Flask(__name__,template_folder='templates_2')
app.config['SECRET_KEY'] = 'your_secret_key_here'


@app.route('/', methods=['GET', 'POST'])
def index():
    if request.method == 'POST':
        # get the uploaded image file from the form
        graph_cnn = load_model("model/graph_model_weights.hdf5")
        file = request.files['image']
        if file:
            filename = secure_filename(file.filename)
            # create the directory if it does not exist
            os.makedirs('static/uploads', exist_ok=True)
            file_path = os.path.join('static/uploads', filename)
            file.save(file_path)
            image = cv2.imread(file_path)
            img = cv2.resize(image, (32,32))
            im2arr = np.array(img)
            im2arr = im2arr.reshape(1,32,32,3)
            img = np.asarray(im2arr)
            img = img.astype('float32')
            img = img/255
            preds = graph_cnn.predict(img)
            predict = np.argmax(preds)
            score = np.amax(preds)
            label = labels[predict]
            # generate URL for uploaded image file
            image_url = url_for('static', filename=f'uploads/{filename}')
            return render_template('result.html', label=label, score=score,image_url=image_url)
    return render_template('index.html')

if __name__ == '__main__':
    app.run(debug=False)
```

Appendix B

Dentistry

B.1 Mouth Ulcer

Mouth ulcers, also known as canker sores, are painful and uncomfortable sores that can appear on the inside of the mouth. They are typically round or oval-shaped with a red, inflamed border and a white or yellowish center. While the exact cause of mouth ulcers is not known, they can be triggered by a variety of factors including stress, certain foods, hormonal changes, and underlying health conditions.

There are three types of mouth ulcers: minor, major, and herpetiform. Minor ulcers are the most common and usually heal within one to two weeks. Major ulcers are larger and deeper and can take several weeks to heal. Herpetiform ulcers are clusters of small ulcers that can be very painful and can take several weeks to heal.

Mouth ulcers can be treated with over-the-counter topical medications such as oral gels or mouthwashes that contain anesthetic or anti-inflammatory agents. In more severe cases, a doctor may prescribe a corticosteroid medication or an antibiotic. It is also important to avoid foods that may irritate

the ulcers and practice good oral hygiene.

Preventing mouth ulcers can be done by avoiding triggers such as stress, acidic or spicy foods, and maintaining good oral hygiene. It is also important to manage any underlying health conditions that may contribute to the development of mouth ulcers.

In conclusion, while mouth ulcers can be painful and uncomfortable, they are typically not a serious health concern and can be effectively treated with over-the-counter or prescription medications. By taking steps to prevent them, individuals can reduce their risk of developing mouth ulcers in the future.

B.2 Cavities

A cavity, also known as tooth decay, is a common dental problem that occurs when the enamel on a tooth is destroyed by acid produced by bacteria in the mouth. Cavities can develop in anyone, but they are more common in children, teenagers, and older adults.

The process of cavity formation starts when bacteria in the mouth produce acid that attacks the enamel on a tooth. The acid causes the enamel to break down, and over time, a hole or cavity can form in the tooth. If left untreated, the cavity can progress deeper into the tooth and eventually reach the inner layers, including the pulp and nerve, leading to pain and infection.

There are several factors that can increase the risk of developing cavities, including poor oral hygiene, a diet high in sugar and carbohydrates, dry mouth, and certain medical conditions that reduce saliva production or increase acid reflux.

Symptoms of a cavity may include:

- Tooth sensitivity to hot, cold, or sweet foods and drinks
- Pain when biting or chewing
- Visible holes or pits in the teeth
- Discoloration or darkening of the tooth

To prevent cavities, it is important to practice good oral hygiene, including brushing teeth twice a day with fluoride toothpaste, flossing daily, and visiting the dentist regularly for check-ups and cleanings. Eating a healthy diet low in sugar and carbohydrates can also help prevent cavities.

If a cavity is detected, treatment options may include filling the cavity with a dental filling, applying a dental crown to protect the tooth, or, in severe cases, root canal therapy to remove the infected pulp and nerve. In some cases, the tooth may need to be extracted if the damage is too severe.

B.3 Gingivitis

Gingivitis is a common gum disease characterized by inflammation of the gums. It is caused by the buildup of plaque on teeth, which is a sticky film of bacteria that forms on teeth and gum lines. When plaque is not removed regularly through brushing and flossing, it can irritate the gums and lead to gingivitis.

The symptoms of gingivitis include red, swollen, and tender gums that bleed easily, especially during brushing or flossing. Additionally, people with gingivitis may experience bad breath or a bad taste in their mouth.

If left untreated, gingivitis can progress into a more serious form of gum disease called periodontitis, which can lead to tooth loss and other health

problems. Therefore, it is important to seek treatment for gingivitis as soon as possible.

Treatment for gingivitis usually involves a professional cleaning by a dentist or dental hygienist to remove plaque and tartar buildup. The patient is also advised to brush their teeth twice a day and floss at least once a day to remove plaque and food particles from the teeth and gums. Additionally, the dentist may recommend a mouthwash or prescribe antibiotics to help clear up the infection.

Prevention is also key in avoiding gingivitis. This can be achieved by practicing good oral hygiene, such as brushing and flossing regularly, using an antiseptic mouthwash, and visiting the dentist for regular check-ups and cleanings. Additionally, maintaining a healthy diet and avoiding tobacco use can help prevent gum disease.

B.4 Tooth Discoloration

Tooth discoloration is a common dental problem that affects the appearance of teeth. It occurs when the color of the tooth changes from its natural shade to a darker or yellowish color. There are several reasons why teeth can become discolored:

1. Genetics: Some people are more prone to tooth discoloration due to their genetics.

2. Aging: As we age, the enamel on our teeth can become thinner, making the underlying dentin more visible. Dentin is naturally yellowish in color, so as it becomes more visible, teeth can appear more yellow.

3. Poor dental hygiene: Neglecting to brush and floss regularly can cause

a buildup of plaque and tartar on the teeth, which can lead to discoloration.

4. Food and drink: Certain foods and drinks can stain teeth over time, especially if consumed regularly. Coffee, tea, red wine, and dark-colored berries are all known to cause tooth discoloration.

5. Tobacco use: Smoking or using other forms of tobacco can cause yellowing of the teeth.

6. Medications: Certain medications can cause tooth discoloration as a side effect. For example, some antibiotics can cause yellowing of the teeth in children.

There are several ways to treat tooth discoloration, depending on the cause and severity of the discoloration. These include:

1. Professional teeth whitening: This is a common treatment for tooth discoloration. It involves using a special whitening solution to remove stains and brighten the teeth.

2. Dental bonding: This involves applying a tooth-colored resin to the surface of the teeth to cover up any discoloration.

3. Porcelain veneers: These are thin shells of porcelain that are bonded to the front of the teeth to cover up any discoloration.

4. Crowns: If the discoloration is severe, a dental crown may be necessary to cover up the tooth and improve its appearance.

Prevention is the best way to avoid tooth discoloration. This includes practicing good oral hygiene, avoiding tobacco use, and limiting consumption

of foods and drinks that can stain teeth. It's also a good idea to schedule regular dental cleanings and checkups with your dentist to keep your teeth healthy and free of discoloration.