**PROJECT REPORT**

**LUNG CANCER DETECTION USING 3D CONVOLUTION NEURAL NETWORKS**

**PROBLEM STATEMENT**

The project deals with the identification of lung cancer, which affects roughly about 225,000 people every year and accounts to about 12 billion in health care costs in the final stages. Therefore, early detection of cancer is critical to give the patients the best chance of recovery and survival.

The goal of this project is to evaluate the data (Slices of CT scans) provided with various pre-processing techniques and analyze the data using machine learning algorithms, in this case 3D Convolutional Neural Networks to train and validate the model, to create an accurate model which can be used to determine whether a person has cancer or not. This will greatly help in the identification and elimination of cancer cells in the early stages.

Therefore, an automated method capable of determining whether the patient will be diagnosed with lung cancer is the aim of this project.

1. **DATASET INFORMATION**

The dataset consists of thousands of CT images from high risk patients in DICOM format (detailed information given below). Each image contains a series with multiple axial slices of the chest cavity and a variable number of 2D slices, which can vary based on the machine taking the scan and the patient.

**DICOM STANDARD**

DICOM is an acronym for Digital Imaging and Communications in Medicine. Files in this format are saved with either a DCM or DCM30 file extension, but some may not have an extension at all. It is used for both communication protocol and a file format i.e., it can store medical information, such as ultrasound and MRI images along with the patient’s information, all in one file.

The format ensures that all the data stays together and provides the ability to transfer information between devices that support the DICOM format. To view the information in the files there are two methods, they are,

1. **Opening DICOM files with Free viewer**

I used the MicroDicom File viewer, which is a free to use software exclusively for viewing the DICOM files. The image along with the metadata can be found making it easy and efficient to access the data for research and development purposes.

1. **Using Libraries to extract the information**

The DICOM standard is complex and there are number of different tools to work with DICOM files, I have used the pydicom package, which is a package that can be imported towards working with images in python.

**FILE DESCRIPTION**

Each patient ID has an associated directory of DICOM files and is in the header of the file along with the patient name, the exact number of images per person varies with every file.

stage1.7z - contains all images for the first stage of the competition, including both the training and test set.

stage1\_labels.csv - contains the cancer ground truth for the stage 1 training set images.

data\_password.txt - contains the decryption key for the image files.

1. **IMPLEMENTATION OVERVIEW**

**Setup process**

• Anaconda Navigator – I used Anaconda Navigator to use Jupyter notebook for python programming.

• NVidia CUDA – Tool used to apply on the GPU for increased and better performance of the system, can only be used on machines having NVidia graphics.

• Installed Tensorflow GPU version on Jupyter Notebook to run the neural network.

• Installed Pydicom for reading the DICOM (Digital Imaging and Communications in Medicine) files.

• Installed OpenCV for manipulating the image like resizing the image.

• Installed MatPlotLib for visualization of the image files and various presentation purposes.

• Installed pandas (used for reading operations on csv files and computation of confusion matrix), os (File and directory manipulations), scikit-learn (to compute confusion matrix) etc.,

After project selection, I decided the technologies to work with and after excessive research and testing, I could zero out on the above technologies. I installed anaconda with python version 3.5 as it supports a variety of functions over the other versions, then I downloaded and setup NVIDIA CUDA on the system, since the size of the processes were large, it was a time intensive procedure to perform with the existing CPU, hence I used CUDA to make use of the GPU for processing and analyzing purposes leading to efficient processing of the datasets. There were problems while setting up CUDA as multiple library files were to be placed exactly in certain locations for it to work properly.

This is my first project using tensorflow and hence importing and using tflearn packages was a challenge. Finally, I could run tensorflow and use the libraries provided. Other libraries were installed and used to complete the project and the description of those libraries are mentioned above.

**Working Process**

The working structure is simple, it consists of three major steps, they are

1. Data Pre-Processing
2. Computation of data via Neural network using 3D-CNN
3. Computation of confusion matrix and cancer prediction

The working structure of the project demanded learning concepts of CNN and going through research papers to get a comprehensive understanding of the activities involved in processing, training and testing a neural network.

1. **Data Pre-Processing**

Let’s look at the data that I am going to work with, and to do so I have used a DICOM file reader to learn about the data. Therefore, with the above data I can understand what the dataset consisted and how I can use it to solve the problem statement. The patient ID and name is provided for every patient in every scan in the folder and ID acts as a unique identifier between the patients.

Taking more metadata tags into consideration I can find a few anomalies which are potential causes for pre-processing. Example: Irregularities in the size of the x and y axis of the image, Depth of the image dependent on the number of axial files available in the folder, sometimes it was the same and sometimes not. The location of one single scan of the entire 3D lung scan was also given for better understanding about which segment of the scan I was dealing with.

This section deals with image pre-processing in specific, as tedious procedures were undertaken to merge these individual slices into 3D images so that I can feed them into a 3D Convolutional neural network. Since I am dealing with large dataset (160GB) it is time intensive and slow in computation

Using the libraries pydicom, matplotlib and OpenCV I can read, visualize and resize the images to the specified format suitable based on the computational speeds and capacity of the system.

1. **3D CNN Computation**

This section consists of the steps involved taking the processed data and using it to determine the accuracy of the model, to predict whether a person could be affected with cancer or not. To understand the composition of what goes into creating a Convolutional neural network and the layers that are used to process the data, a summary of every layer and its functions that I used in the project is provided, the summary is an amalgamation of inferences that I achieved based on various research papers whose links are also provided (references).

The 3D images given as input are pre-processed and various layers are set with values, there is no hard and fast rule that values are fixed, hence I had to test the layers’ multiple times with different values and arrive at a set of values that were optimal.

So, when I train the model, I send in a part of the data (majority) and keep the rest of it for validation purposes to check is the model is well trained. I trained 1595 images in total, out of which 1495 images as input for training the model and 100 images for validation.

The idea here is to complete training the model with 1495 images and then pass the 100 validation images to test the model to provide an accuracy as to how well the model is trained and can predict accurately. In short, I am providing the test data on the trained data model to identify the accuracy (final accuracy).

The number of epochs to run is set, batch size of the input data specifying how many images are to be processed at once, input size of the image was also specified as 50x50x20, it is very small compared to the original size of the image, but our systems could accommodate only such capacity, this in turn caused a drop in the quality and quantity of the dataset.

This drop-in metrics lead us to observe that I am not using enough information for the model to train on and hence affect the accuracy of the model. After running the model for 100 epochs, I found that the percentage of accuracy was stagnant at the 83rd epoch at around 72%, which is the maximum accuracy I could achieve, I could not achieve more due to lack of more data causing the model to overfit, leading to lower accuracy. Before identifying the final accuracy, I tried to mix and match the layers in between to get the optimal layer composition, i.e., At first I set 3 layers and achieved an accuracy of 54% with just 3 epochs run, then I increased the number of epochs and observed that the accuracy increased, I kept increasing the layers and finally reached 6 layers which was optimal and 100 epochs was just fine to get the maximum accuracy.

1. **Computation of confusion matrix and cancer prediction**

The confusion matrix is a way to determine whether the model is good or not. Here I implemented confusion matrix for predicted labels vs actual labels. The 0’s represent no cancer and the 1’s represent cancer. Ideally the false positives and the false negatives value should be low.

**ERRORS FACED DURING EXECUTION**

I specified most of the outcomes from the implementation overview, but when it comes to the CNN process, there were many challenges faced

1. **Resizing the z axis due to irregularities in dimensions**

I noted that for each patient the number of slices varied. Feeding patient with various slices (z axis) to the neural network would pose a problem and would not give us the desired results. So, I must chunk the slices and make them smaller and equal.

1. **Tensor reshape error**

Tensor automatically throws a reshape error every time the data is fed to the neural network and this would pose as a problem. I passed this so that tensor can take the shape specified by me

1. **Resource exhausted error**

Tensorflow was using the GPU at the back and every time the neural network ran, so the kernel had to be restarted to free up the GPU’s memory. When trying to feed a larger image size, Iran into the same error. This was because there was no sufficient memory to process such a huge dataset

1. **Dataset is not enough – overfitting occurs**

The total dataset size is 1595 out of which 1107 patients do not have cancer, 390 patients have cancer and 98 patients do not have labels. The dataset for number of patients having cancer is very low. This is the reason why overfitting occurs and cancer predictions are not that accurate. One way to overcome this problem is by adding more cancer patient datasets which is currently not available.

**CONCLUSION**

The above project was a learning experience for me to implement Convolutional neural nets for 3D images. I learnt image processing and gained knowledge on resizing and processing large number of images using python packages. Working with large dataset was a challenge but implementing tensorflow gpu made it easier to train 3D models. Encountering problems such as overfitting and resource exhausted errors helped me to handle and prevent them from happening again.

**FUTURE SCOPE**

The idea that I have presented is at the very beginning of its deployment phase and thus I wish to extend my project by implementing a model that can make valuable predictions based on the reports specified in the datasets and achieve a higher value of accuracy by using Microsoft’s ResNet approach to build a network, to provide much higher learning rate of the nodes increasing the performance of the model and minimizing the losses effectively. I can move forward with the deployment process making this application ground breaking regarding the medical industry, improving the health care system thus causing a huge beneficial impact on the lives of many individuals.