A Hybrid Diagnostic System: Integrating Rule-Based Expert Systems and Deep Learning for Enhanced Medical Decision-Making

CS 3642 Section W01 (Fall 2024)

Shreya Katare
College of Computing and Software Engineering
Kennesaw State University
Marietta, GA
skatare

Katherine Smith
College of Computing and Software Engineering
Kennesaw State University
Marietta, GA
ksmit843

Abstract — This project has the goal of implementing a hybrid approach of rule-based expert systems and deep learning in medical diagnosing. It uses two datasets of lung cancer, one that describe various patient profiles and one that shows associated lung CT scan images. Both datasets are extracted from Kaggle. The system aims to diagnose symptoms based on a written profile of a patient with a rule-based algorithm. If it is undiagnosable with the given conditions or is ambiguous, then the system moves on to diagnosis with deep learning. This would relay further information about whether the patient has lung cancer. The system also keeps track of accuracy of the actual diagnoses compared to the ones given by the system, increasing as the system improves. This project will be implemented within the Visual Studio Code platform and coded in the Python programming language. Additionally, storage and collaboration of the code is done with GitHub. This hybrid approach aims to enhance future medical diagnoses with Artificial Intelligence.

Keywords—artificial intelligence, deep learning, image analysis, knowledge-based, lung cancer, medical diagnosis, rule-based.

I. BACKGROUND INFORMATION

This project focuses on rule-based expert systems and deep learning for medical diagnosis. Healthcare professionals for years have been using their medical knowledge to properly diagnose diseases and various illnesses. They use the knowledge they have to make proper diagnoses based on symptoms and observations. However, with technology growing there have been new ideas on how to improve and help healthcare providers to make proper diagnoses. Two of these technologies that have been studied and used are rule-based expert systems and deep learning for medical diagnosis.

Rule-based expert systems have been around longer than the new deep learning models that people see today. They use sets of if-then statements to make accurate predictions. This can be helpful in the medical field for multiple reasons. Doctors use information to make proper diagnoses but just like any professional there can be mistakes. A rule-based expert system is a great solution to this problem as it provides accurate answers to these problems. Helping doctors to work faster and more efficiently on the job. Being able to check over their own predictions or helping to make faster decisions on the field when handling multiple patients. This is what makes these expert systems so appealing to medical professionals and why they are still being studied and improved upon today.

Deep learning models can be used in many different ways in the medical field. "DL-based algorithms, using architectures such as convolutional neural networks (CNNs), are distinct from traditional machine learning approaches." [6]. CNNs will be our main focus for this research. CNNs or convolutional neural networks are a set of nodes that are connected to make a neural network. With a convolution layer, pooling layer, and fully connected layer. These CNN models are first trained and then able to take on vast amounts of data and find patterns within the data to come up with predictions for the users. This is another reason why CNNs are also popular in the medical field. The ability for these models to find patterns and make predictions is exactly what is needed in the field. However, these CNNs are mostly used for medical imaging. This is because CNNs are great at taking in medical images and finding patterns between the images to make proper diagnoses on images such as CT scans or MRI scans. For this paper we will begin to discuss these models and how they can be used alongside rule-based expert systems for medical diagnosis.

II. PROBLEM STATEMENT

The increasing rise of Artificial Intelligence in various fields of society has caused the need for perfect accuracy. One of the fields that particularly need this precision is medical, particularly in its diagnoses. There already exists many rule-based systems for medical diagnosis based on patient profiles. Additionally, medical diagnosis with deep learning is also not a new concept. However, these systems have problems in accuracy when there are ambiguous cases as they are individually run.

This project brings forth the idea of combining these two expert systems into one. In particular, creating a hybrid approach of rule-based and deep learning is the aim of this project. It will first analyze patient data of their symptoms with rule based. If it turns out with an ambiguous diagnosis, then the system moves on to the deep learning component. Here, patient images of their lungs are analyzed and spotted whether they have tumors.

The problem this hybrid system seeks to solve is combining rule-based and deep learning-based systems for better lung cancer analysis. By combining both of the approaches, the system will give out more reliable diagnostic outcomes while steadily improving accuracy as well.

III. LITERATURE REVIEWS

A. Rule-based Expert Systems

So far, there have been multiple implementations of the rule-based expert systems. Many of them are done with patient data as explained in Gulavani and Kulkarni's journal article. The applications are various: INTERNIST, MYCIN, CADUCEUS, QMR-Quick Medical Reference, PUFF-Pulmonary Function System, ATHENA, CEMS, ERA-Early Referrals Application, GIDEON-Global Infectious Disease and Epidemiology Network, PERFEX-Knowledge Based Interpretation of Myocardial SPECT Imagery, Iliad, Isabel, LISA, Expert System for the Diagnosis of neonatal jaundice for use by medical field personnel, and MEDUSA. They are often called as "knowledge-based" because of their reliance on the pre-defined rules provided with the artificial intelligence. Medical diagnosis, in particular, is often composed of a "knowledge base, inference engine, knowledge acquisition, explanation facility, and user interface" [1]. These are required to fulfill the goal of an expert medical diagnostic system, which is to have the best accuracy score.

A similar description of the requirements are described in Painuli's journal article. They include:

a. a natural language (such as English) to interface and interact with the user

b. a knowledge base containing the rules from which the decisions can be made

c. a database of facts specific to the domain of focus

d. an inference engine to solve problems by linking the knowledge base rules with the database, using heuristics or "rules of thumb" logic

These conditions allow for an expert system to work effectively and determines whether it is rule-based. The article also explains that there are major issues within the field of autoimmune disease [2]. Diagnosing in this subject is essential for better accuracy, persuading for the lung cancer dataset used for this project.

A particular display of the rule-based system is described in Cincar and Ivașcu's conference paper. It monitors patients in real-time, noting their health statuses, and uses a Metabolic Equivalent of Tasks (MET) to determine their overall health condition. The rule-based component in this system is matching facts to determine a patient's health condition. The simulation yields successful results since they are accurate to the actual results and the proposed system "automatically computes the threshold ranges and creates the rules for each monitored vital sign for each physical activity known by the system" [3]. This and many other examples given in this project have the goal to predict an accurate diagnosis.

This project will also use predictive rule-based expert systems. As described in Kattan's journal article, these sorts of medical expert systems (MES) are "strictly making predictions of some medical endpoint or outcome that is not immediately known" as compared to the prescriptive MES [4]. Further enhancement in the future projects within the MES for rule-based may be done by incorporating the prescriptive component.

B. Deep Learning Systems

Deep learning systems have been making major improvements for the past few years in medical diagnosis. They provide insight into what the future may hold for technology and medicine. Key areas that this research will look into include CNNs and neural networks. Machine learning has offered a way for us to create technology that can view and analyze data as a human would, however, faster and better.

There are a few different types of deep learning systems that all have various benefits. The top 3 are Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Generative Adversarial Networks (GANs) [7]. The studies that this paper looks into have deep significance into our own research. They go into how deep learning is being used in various ways for medical diagnosis.

The proposed methodology for Dahwan [8] was to use CNNs to predict whether MRI scans contained tumors or not. This study was done on a set of data containing MRI scans where the CNNs were trained and then able to scan images to make predictions on the scans. The main algorithm they used EfficientNet v2-S, ended with an accuracy of 98.03% according to their table [8]. Using CNNs for image scanning has far beyond proven its ability to work for medical imaging. CNNs are designed to scan images in smaller parts, detect smaller patterns then combine for an overall output at the final layer. The CNN captures images in small parts and can scan multiple edges and textures in each individual neuron which are then combined to create the neural network that makes the final classification.

Awotunde's research was done using a hybrid system with rule-based hybrid feature selection mechanisms that eliminate features that are deemed irrelevant, then a deep learning system that helps to diagnosis breast cancer [9]. This type of hybrid system is what this research paper intends to go in depth with. By using rule-based expert systems for initial input, predicting on obvious features, the deep learning model is then able to take the rest of the input for a proper diagnosis for more difficult cases. The system that Awotunde used showed an accuracy of 99.5% accuracy [9]. A significantly high accuracy for a system that was tested on a vast dataset.

Each of the various studies has their own gaps that still can be improved upon. Deep learning systems such as these have only just begun showing signs of their capabilities. Combined with an accurate rule-based system, as shown from Awotunde's research, there may be a possibility to improve upon these systems.

IV. PROPOSED ALGORITHMS AND APPLICATIONS

The algorithms used for this project include the rule-based system and the deep learning component. The rule-based algorithm processes patient symptoms to diagnose lung cancer. The deep learning algorithm analyzes cases that were deemed too complex for the rule-based system. This hybridization ensures that straightforward cases are handled efficiently, while more complex cases requiring more pattern recognition are handled by the deep learning model. Additionally, an accuracy score is kept track throughout the code.

The rule-based system in particular relies on a pre-defined set of "instructions" that are based on a dataset of lung cancer symptoms. The system evaluates the patient's input against these rules to provide a diagnosis or flag cases for deep learning. The input for this algorithm include symptoms such as smoking, yellow fingers, anxiety, peer pressure, chronic disease, fatigue, allergy, wheezing, alcohol consumption, coughing, shortness of breath, swallowing difficulty, and chest pain. For rule matching to work in this part of the project, each rule in the system represents a diagnosis and its corresponding symptoms. For example, there are rules for diagnosing lung cancer, diagnosing no lung cancer, and diagnosing non-determinable. The system checks if there are any matching symptoms according to the "&&" and "||" statements.

The deep learning system is responsible for analyzing cases that were deemed more complex. In particular, when the rule-based system cannot make a confident diagnosis, this model is invoked to assist in complex decision-making. Input in this algorithm requires data from the same patients that were flagged as "non-determinable" in the rule-based system. The deep learning system requires that labels and features are set. For this model the features were the symptoms of the patients, and the labels were the dataset's predisposed diagnosis of lung cancer. With this the system is able to train before compiling and making predictions. It is made with multiple layers all with set neurons to analyze the data and find patterns. The data is first sent through the first layer where each feature adds a new neuron to the system. The second layer finds patterns between these neurons. Finally, the output layer which makes the prediction.

V. IMPLEMENTATION AND EXPERIMENTAL RESULTS

A) Rule-based Expert Systems

The rule-based component of the hybrid medical diagnosis system was coded in the Python programming language. It includes a dataset of patients' symptoms and their real diagnosis. The data table used for this project is in the form of a csv file and is first imported. It contains a total of 310 patient cases, each with its own YES/NO symptoms. These symptoms are smoking, yellow fingers, anxiety, peer pressure, chronic disease, fatigue, allergy, wheezing, alcohol consumption, coughing, shortness of breath, swallowing difficulty, and chest pain. Before the rules are created, preprocessing the data is required. It includes handling missing values in the table, stripping the columns so that there are no empty spaces, and converting rows into facts. The last part is used to make sure there are developer-friendly column names. The code snippet below demonstrates these pre-processing steps:

```
# load csv file

df = pd.read_csv('lung cancer survey.csv')
```

```
# handling missing values

df.fillna({"SMOKING": 1, "YELLOW_FINGERS": 1, "ANXIETY": 1, "PEER_PRESSURE": 1,

        "CHRONIC DISEASE": 1, "FATIGUE": 1, "ALLERGY": 1, "WHEEZING": 1,

        "ALCOHOL CONSUMING": 1, "COUGHING": 1, "SHORTNESS OF BREATH": 1,

        "SWALLOWING DIFFICULTY": 1, "CHEST PAIN": 1, "LUNG CANCER": "NO"}, inplace=True)

df.columns = df.columns.str.strip()

# converting rows of dataset into facts

def process_row_to_facts(row):

    return {

        "SMOKING": row["SMOKING"],

        "YELLOW_FINGERS": row["YELLOW_FINGERS"],

        "ANXIETY": row["ANXIETY"],

        "PEER_PRESSURE": row["PEER_PRESSURE"],

        "CHRONIC_DISEASE": row["CHRONIC DISEASE"],

        "FATIGUE": row["FATIGUE"],

        "ALLERGY": row["ALLERGY"],

        "WHEEZING": row["WHEEZING"],

        "ALCOHOL_CONSUMING": row["ALCOHOL CONSUMING"],

        "COUGHING": row["COUGHING"],

        "SHORTNESS_BREATH": row["SHORTNESS OF BREATH"],

        "SWALLOWING_DIFFICULTY": row["SWALLOWING DIFFICULTY"],

        "CHEST_PAIN": row["CHEST PAIN"],

        "LUNG_CANCER": row["LUNG_CANCER"]

    }
```

When pre-processing is done, the next step is to define the rules for the system. For this, a RuleBasedSystem class is made, defining the methods for initialization, adding rules, evaluation, action functions, and rule conditions. The add_rule method adds rules in a (condition, action) format in a rules list. The evaluate function compares the facts against the rules and triggers the appropriate diagnosis. Below are the following code snippets for these methods:

```
# define the rule-based system

class RuleBasedSystem:

    def __init__(self):

        self.rules = []

    def add_rule(self, condition, action):

        # adds a new rule (condition, action) to the system

        self.rules.append((condition, action))

    def evaluate(self, facts):

        # evaluates the facts against the rules and triggers the appropriate diagnosis

        for condition, action in self.rules:

            if condition(facts):

                return action(facts)

        return "No diagnosis made"
```

There are three rule conditions, and their associated diagnoses also used in the class: cancer, no cancer, and no diagnosis. The cancer conditions are triggered if the patient has smoking, coughing, or chest pain, and wheezing. These symptoms of lung cancer were determined from the MedlinePlus website supported by the National Library of Medicine. It determines that “chest pain or discomfort, a cough that doesn’t go away or gets worse over time, coughing up blood, trouble breathing, wheezing, hoarseness, loss of appetite, weight loss for no known reason, feeling very tired, trouble swallowing, and swelling in the face and/or veins in the neck” are common for lung cancer [5]. If true, the patient is diagnosed with “Lung Cancer Diagnosed.” The no cancer conditions are triggered if the patient has mild symptoms such as no smoking, no coughing, no chest pain, and no wheezing. If true, the patient is diagnosed with “No Lung Cancer.” The ambiguous rule conditions are triggered if there are mixed symptoms of smoking, no coughing, no chest pain, and shortness of breath. If true, the patient is diagnosed with “No diagnosis made" and moved to the deep learning component of this project. Below are the associated code snippets:

```
# action functions

def diagnose_lung_cancer(facts):

   return "Lung Cancer Diagnosed"

def diagnose_no_cancer(facts):

   return "No Lung Cancer"

# rule conditions

def rule_lung_cancer(facts):

   # triggers when certain symptoms are present

   return (facts["SMOKING"] == 2 and
(facts["COUGHING"] == 2 or facts["CHEST_PAIN"] ==
2) and

         facts["WHEEZING"] == 2)

def rule_no_lung_cancer(facts):

   # triggers when symptoms are mild

   return (facts["SMOKING"] == 1 and
facts["COUGHING"] == 1 and

         facts["CHEST_PAIN"] == 1 and
facts["WHEEZING"] == 1)

def rule_ambiguous(facts):

   # triggers when symptoms are mixed or ambiguous

   return (facts["SMOKING"] == 2 and
facts["COUGHING"] == 1 and

         facts["CHEST_PAIN"] == 1 and
facts["SHORTNESS_BREATH"] == 2)

# create the rule-based system

system = RuleBasedSystem()

# add rules

system.add_rule(rule_lung_cancer, diagnose_lung_cancer)

system.add_rule(rule_no_lung_cancer,
diagnose_no_cancer)

system.add_rule(rule_ambiguous, lambda facts: "No
diagnosis made")
```

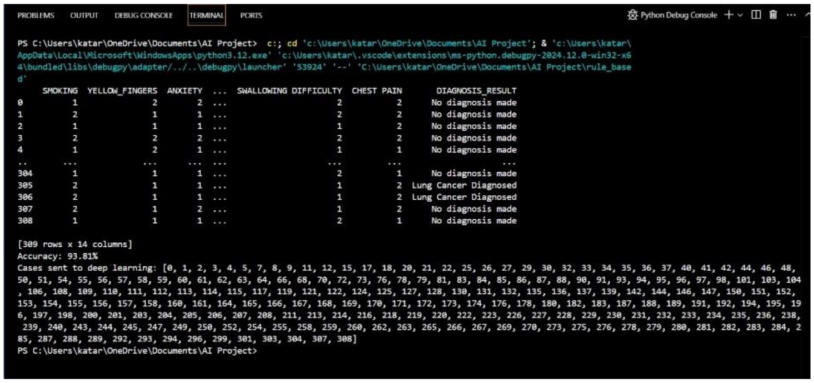
Additionally, the cases sent to deep learning are done within a for loop and the number of correct predictions are kept track. The for loop determines if any row in the original data table is diagnosed with "No diagnosis made." If true, the case is appended to the deep_learning_cases list. If false, the correct result is compared with the system-driven diagnosis. If they are the same, correct_predictions adds the case. Below are the code snippets for these methods:

```
   # apply the rules to each row in the dataset
   results = []
   deep_learning_cases = [] # collect cases with "No
diagosis made" for deep learning
   correct_predictions = 0
   total_predictions = 0

   for index, row in df.iterrows():
      facts = process_row_to_facts(row)
      diagnosis = system.evaluate(facts)

      # if no diagnosis is made, flag for deep learning
      if diagnosis == "No diagnosis made":
         deep_learning_cases.append(index)  # collect cases
for deep learning
      else:
         # compare the diagnosis with actual lung cancer
result
         actual = row["LUNG_CANCER"]
         if (diagnosis == "Lung Cancer Diagnosed" and
actual == "YES") or \
            (diagnosis == "No Lung Cancer" and actual ==
"NO"):
            correct_predictions += 1
         total_predictions += 1

      results.append(diagnosis)
```

The last component needed in the rule-based program is outputting the diagnosis results in the console and calculating the accuracy. Below is the implementation for this part:

```
   # output the diagnosis results
   df["DIAGNOSIS_RESULT"] = results
   print(df[["SMOKING", "YELLOW_FINGERS",
"ANXIETY", "PEER_PRESSURE", "CHRONIC
DISEASE", "FATIGUE", "ALLERGY",
            "WHEEZING", "ALCOHOL CONSUMING",
"COUGHING", "SHORTNESS OF BREATH",
"SWALLOWING DIFFICULTY",
            "CHEST PAIN", "DIAGNOSIS_RESULT"]])
   # calculate accuracy
   if total_predictions > 0:
        accuracy = correct_predictions / total_predictions
        print(f"Accuracy: {accuracy:.2%}")
    else:
        print("No cases diagnosed to calculate accuracy.")
    # print cases that need deep learning for further analysis
    print(f"Cases sent to deep learning:
{deep_learning_cases}")
```

Below is the associated screenshot for the rule-based component of this project:



```
PS C:\Users\katar\OneDrive\Documents\AI Project>  c:; cd 'c:\Users\katar\OneDrive\Documents\AI Project'; & 'c:\Users\katar\
AppData\Local\Microsoft\WindowsApps\python3.12.exe' 'c:\Users\katar\.vscode\extensions\ms-python.debugpy-2024.12.0-win32-x6
4\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '53924' '--' 'C:\Users\katar\OneDrive\Documents\AI Project\rule_base
d'
     SMOKING  YELLOW_FINGERS  ANXIETY  ...  SWALLOWING DIFFICULTY  CHEST PAIN      DIAGNOSIS_RESULT
0          1               2        2  ...                      2           2      No diagnosis made
1          2               1        1  ...                      2           2      No diagnosis made
2          1               1        1  ...                      1           2      No diagnosis made
3          2               2        2  ...                      2           2      No diagnosis made
4          1               2        1  ...                      1           1      No diagnosis made
..       ...             ...      ...  ...                    ...         ...                    ...
304        1               1        1  ...                      2           1      No diagnosis made
305        2               1        1  ...                      1           2  Lung Cancer Diagnosed
306        2               1        1  ...                      1           2  Lung Cancer Diagnosed
307        2               1        2  ...                      1           2      No diagnosis made
308        1               1        1  ...                      2           1      No diagnosis made

[309 rows x 14 columns]
Accuracy: 93.81%
Cases sent to deep learning: [0, 1, 2, 3, 4, 5, 7, 8, 9, 11, 12, 15, 17, 18, 20, 21, 22, 25, 26, 27, 29, 30, 32, 33, 34, 35, 36, 37, 40, 41, 42, 44, 46, 48,
50, 51, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 66, 68, 70, 72, 73, 76, 78, 79, 81, 83, 84, 85, 86, 87, 88, 90, 91, 93, 94, 95, 96, 97, 98, 101, 103, 104
, 106, 108, 109, 110, 111, 112, 113, 114, 115, 117, 119, 121, 122, 124, 125, 127, 128, 130, 131, 132, 135, 136, 137, 139, 142, 144, 146, 147, 150, 151, 152,
153, 154, 155, 156, 157, 158, 160, 161, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 176, 178, 180, 182, 183, 187, 188, 189, 191, 192, 194, 195, 19
6, 197, 198, 200, 201, 203, 204, 205, 206, 207, 208, 211, 213, 214, 216, 218, 219, 220, 222, 223, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 238,
 239, 240, 243, 244, 245, 247, 249, 250, 252, 254, 255, 258, 259, 260, 262, 263, 265, 266, 267, 269, 270, 273, 275, 276, 278, 279, 280, 281, 282, 283, 284, 2
85, 287, 288, 289, 292, 293, 294, 296, 299, 301, 303, 304, 307, 308]
PS C:\Users\katar\OneDrive\Documents\AI Project>
```

Figure 1: Console output for rule-based system

Note that the accuracy score is fairly high at 93.91%.

B. Deep Learning Systems

The deep learning cases that were collected are then evaluated by the deep learning model. For this system there is a separate dataset made to collect all the cases labeled “No diagnosis made” under “DIAGNOSIS_RESULT”. These cases are then copied to a new dataset labeled “deep_learning_df”. The YES/NO responses from the original data frame are changed to True/False which is then changed to 0/1 for more efficiency with the system. The dataset is then separated into features and labels, X and y respectively. With X being all the symptoms from the dataset, and y being the 0/1 labels for lung cancer. These are to provide the system with a way to compare and analyze patterns amongst the symptoms that leads to lung cancer. The label encoder transforms the y label into a numerical array for the system.

The next part to setting up the system is the train test split function. This allows the system to split the data into training data and testing data, the “test_size” is set to 0.2 for 20% of the data to be set aside. Then a random number so the split is even each time. The system had an issue with the values in the Lung cancer row. There were 279 true values but only 31 false values. This created a bias and an imbalance which is fixed with class weight. The class weight function creates a balance between the true and false values, to ensure that the system does not predict true values more often than false, for a high accuracy score. Below are the associated code snippets:

```
    df["LUNG_CANCER"] =
df["LUNG_CANCER"].map({"YES":True, "NO":False})
    #Create dataframe with deep learning cases
    deep_learning_df = df[df['DIAGNOSIS_RESULT'] ==
"No diagnosis made"].copy()
    deep_learning_df["LUNG_CANCER"] =
deep_learning_df["LUNG_CANCER"].map({True: 1,
False: 0})

    #deep learning model using tensorflow


    #assigning the features and the the labels
    #X being the values to be tested and y being what the
expected value should be
    X =
deep_learning_df.drop(columns=['DIAGNOSIS_RESULT',
'LUNG_CANCER', 'GENDER', 'AGE'])
    y = deep_learning_df['LUNG_CANCER']
      print(deep_learning_df)


      le = LabelEncoder()
      y = le.fit_transform(y)


    #Splitting the train set and the test set with 25% of data
used for test
    x_train, x_test, y_train, y_test = train_test_split(X, y,
test_size = .25, random_state=42)

    #adding a class weight to cause more loss in the model
if a false value is guessed wrong
    #this is because there are more true values for lung
cancer than false in the dataset which causes the model to
skew towards true
    class_weight = compute_class_weight('balanced',
classes = np.unique(y_train), y =y_train)
    class_weight = dict(enumerate(class_weight))
```

Creating the neural network is the next step in implementing the deep learning model. Each layer is added in using the activation ReLU (Rectified Linear Unit) which is a common non-linear activation method in TensorFlow. The first layer is added in with the input_shape allowing for the correct number of neurons, which is based on the number of symptoms in the dataset (13). The last layer has a singular output which is 1 and activation sigmoid which is used for a lot of binary classification models such as this one. The model is then compiled using Adam and binary crossentropy, with the metrics being accuracy for a score on how the system is doing. The model is then trained with the fit function where both training values are used and the class weight earlier set is used as well. Below is a snippet of this code:

```
    #adding layers to the neural network
    model = Sequential()
    model.add(Dense(units=16, activation='relu',
input_shape=(X.shape[1],)))
    #model.add(Dropout(0.2))
    model.add(Dense(units=8, activation='relu'))
    #model.add(Dropout(0.2))
    model.add(Dense(1,activation='sigmoid'))



    #Preparing the model for training with the optimizer as
Adam and binary crossentropy for the 2 values that the
prediction can be
    model.compile(optimizer='Adam',
loss='binary_crossentropy', metrics=['accuracy'])
    model.summary()


    #training the actual model
    model.fit(x_train,y_train, batch_size=20,
epochs=10,class_weight = class_weight)
```

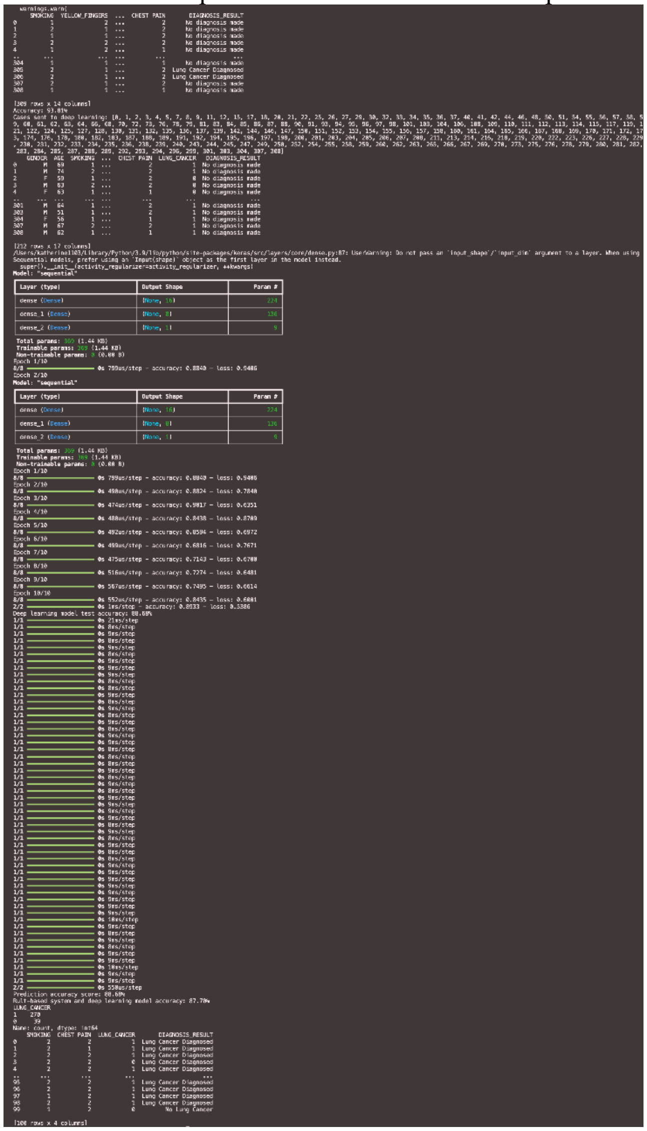
The model is then evaluated for an accuracy score to show how well the test sets did. Then the code is set to predict every individual value in the dataset. There is ability to predict batches of the data which the code shows later, however for this model each one is predicted individually to be added back to the rule-based system. The code for this is listed below:

```
    test_loss, test_accuracy = model.evaluate(x_test, y_test)
    print(f"Deep learning model test accuracy:
{test_accuracy*100:.2f}%")

    #new prediction model for updating the dataset as the
model predicts
    #Normally predictions are made on a portion of the data
set this is so every row has a prediction
    for i in X.index:
    data = X.loc[[i]]

    prediction = model.predict(data)

    predicted = int(prediction[0,0] > 0.5) #based on the
prediction if it is greater than 0.5 predicts 1 otherwise 0

    deep_learning_df.at[i, 'DIAGNOSIS_RESULT'] = 'Lung
Cancer Diagnosed' if predicted == 1 else 'No Lung Cancer'
#adds the predition back into the dataframe

    #Prediction to show accuracy of model
    prediction = model.predict(x_test, batch_size=32)

    predicted = (prediction>0.5).astype(int)

    actual = y_test.flatten()

    #Accuracy of predictions
    ac_score = accuracy_score(actual, predicted)
    print(f"Prediction accuracy score:
{ac_score*100:.2f}%")
```

Finally, for the final part all values in the original dataset containing “No diagnosis made” are dropped from that dataset. The code then adds back to the dataset from the deep learning model where the diagnosis result was collected. The dataset is then calculated again for accuracy of the overall system with the deep learning model and the rule based system. Below is the code:

```
    #drop all rows with no diagnosis made in the original
dataframe
    drop_df = df[df['DIAGNOSIS_RESULT']=="No
diagnosis made"].index
    df.drop(drop_df, inplace=True)

    #add back the deep learning models predictions
    df = pd.concat([df, deep_learning_df],
ignore_index=True)



    #testing the accuracy of the Rule-Based system and deep
learning model from the actual results
    correct_diagnosis = ((df['LUNG_CANCER']==True) &
(df['DIAGNOSIS_RESULT'] == "Lung Cancer
Diagnosed") | (df['LUNG_CANCER']==False) &
(df['DIAGNOSIS_RESULT']=="No Lung Cancer")).sum()
    accuracy = correct_diagnosis/len(df)

    print(f"Rult-based system and deep learning model
accuracy: {accuracy*100:.2f}%")


    print(df['LUNG_CANCER'].value_counts()) #show how
many in dataset have lung cancer and who does not
    # Manually check a few cases
    print(df[['SMOKING', 'CHEST PAIN',
'LUNG_CANCER', 'DIAGNOSIS_RESULT']].head(100))
```

The model’s prints the output of the overall accuracy of the system which averages around 83.95%. The system was run 5 times to gather an average accuracy of the system. The lower than average score could be due to multiple factors regarding the neural network and how it reads the data and finds the patterns. Below is one of the outputs:



```
  warnings.warn(
     SMOKING  YELLOW_FINGERS  ...  CHEST PAIN      DIAGNOSIS_RESULT
0          1               2  ...           2      No diagnosis made
1          2               1  ...           2      No diagnosis made
2          1               1  ...           2      No diagnosis made
3          2               2  ...           2      No diagnosis made
4          1               2  ...           1      No diagnosis made
..       ...             ...  ...         ...                    ...
304        1               1  ...           1      No diagnosis made
305        2               1  ...           2  Lung Cancer Diagnosed
306        2               1  ...           2  Lung Cancer Diagnosed
307        2               1  ...           2      No diagnosis made
308        1               1  ...           1      No diagnosis made

[309 rows x 14 columns]
Accuracy: 93.01%
Cases sent to deep learning: [0, 1, 2, 3, 4, 5, 7, 8, 9, 11, 12, 15, 17, 18, 20, 21, 22, 25, 26, 27, 29, 30, 32, 33, 34, 35, 36, 37, 40, 41, 42, 44, 46, 48, 50, 51, 54, 55, 56, 57, 58, 5
9, 60, 61, 62, 63, 64, 66, 68, 70, 72, 73, 76, 78, 79, 81, 83, 84, 85, 86, 87, 88, 90, 91, 93, 94, 95, 96, 97, 98, 101, 103, 104, 106, 108, 109, 110, 111, 112, 113, 114, 115, 117, 119, 1
21, 122, 124, 125, 127, 128, 130, 131, 132, 135, 136, 137, 139, 142, 144, 146, 147, 150, 151, 152, 153, 154, 155, 156, 157, 158, 160, 161, 164, 165, 166, 167, 168, 169, 170, 171, 172, 17
3, 174, 176, 178, 180, 182, 183, 187, 188, 189, 191, 192, 194, 195, 196, 197, 198, 200, 201, 203, 204, 205, 206, 207, 208, 211, 213, 214, 216, 218, 219, 220, 222, 223, 226, 227, 228, 229
, 230, 231, 232, 233, 234, 235, 236, 238, 239, 240, 243, 244, 245, 247, 249, 250, 252, 254, 255, 258, 259, 260, 262, 263, 265, 266, 267, 269, 270, 273, 275, 276, 278, 279, 280, 281, 282,
 283, 284, 285, 287, 288, 289, 292, 293, 294, 296, 299, 301, 303, 304, 307, 308]
    GENDER  AGE  SMOKING  ...  CHEST PAIN  LUNG_CANCER   DIAGNOSIS_RESULT
0        M   69        1  ...           2            1  No diagnosis made
1        M   74        2  ...           2            1  No diagnosis made
2        F   59        1  ...           2            0  No diagnosis made
3        M   63        2  ...           2            0  No diagnosis made
4        F   63        1  ...           1            0  No diagnosis made
..     ...  ...      ...  ...         ...          ...                ...
301      M   64        1  ...           2            1  No diagnosis made
303      M   51        1  ...           2            1  No diagnosis made
304      F   56        1  ...           1            1  No diagnosis made
307      M   67        2  ...           2            1  No diagnosis made
308      M   62        1  ...           1            1  No diagnosis made

[212 rows x 17 columns]
/Users/katherine1103/Library/Python/3.9/lib/python/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using
Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Model: "sequential"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓
┃ Layer (type)                    ┃ Output Shape           ┃       Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩
│ dense (Dense)                   │ (None, 16)             │           224 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_1 (Dense)                 │ (None, 8)              │           136 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_2 (Dense)                 │ (None, 1)              │             9 │
└─────────────────────────────────┴────────────────────────┴───────────────┘
 Total params: 369 (1.44 KB)
 Trainable params: 369 (1.44 KB)
 Non-trainable params: 0 (0.00 B)
Epoch 1/10
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 799us/step - accuracy: 0.8840 - loss: 0.9486
Epoch 2/10
Model: "sequential"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓
┃ Layer (type)                    ┃ Output Shape           ┃       Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩
│ dense (Dense)                   │ (None, 16)             │           224 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_1 (Dense)                 │ (None, 8)              │           136 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_2 (Dense)                 │ (None, 1)              │             9 │
└─────────────────────────────────┴────────────────────────┴───────────────┘
 Total params: 369 (1.44 KB)
 Trainable params: 369 (1.44 KB)
 Non-trainable params: 0 (0.00 B)
Epoch 1/10
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 799us/step - accuracy: 0.8840 - loss: 0.9486
Epoch 2/10
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 498us/step - accuracy: 0.8824 - loss: 0.7840
Epoch 3/10
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 474us/step - accuracy: 0.9017 - loss: 0.6351
Epoch 4/10
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 488us/step - accuracy: 0.8438 - loss: 0.8709
Epoch 5/10
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 492us/step - accuracy: 0.8594 - loss: 0.6972
Epoch 6/10
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 499us/step - accuracy: 0.6816 - loss: 0.7671
Epoch 7/10
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 475us/step - accuracy: 0.7143 - loss: 0.6700
Epoch 8/10
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 516us/step - accuracy: 0.7274 - loss: 0.6481
Epoch 9/10
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 567us/step - accuracy: 0.7495 - loss: 0.6614
Epoch 10/10
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 552us/step - accuracy: 0.8435 - loss: 0.6001
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step - accuracy: 0.8933 - loss: 0.3386
Deep learning model test accuracy: 88.60%
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 10ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 8ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 10ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
2/2 ━━━━━━━━━━━━━━━━━━━━ 0s 550us/step
Prediction accuracy score: 88.60%
Rult-based system and deep learning model accuracy: 87.70%
LUNG_CANCER
1    270
0     39
Name: count, dtype: int64
    SMOKING  CHEST PAIN  LUNG_CANCER       DIAGNOSIS_RESULT
0         2           2            1  Lung Cancer Diagnosed
1         2           1            1  Lung Cancer Diagnosed
2         2           2            1  Lung Cancer Diagnosed
3         2           2            0  Lung Cancer Diagnosed
4         2           2            1  Lung Cancer Diagnosed
..      ...         ...          ...                    ...
95        2           2            1  Lung Cancer Diagnosed
96        2           2            1  Lung Cancer Diagnosed
97        1           2            1  Lung Cancer Diagnosed
98        2           2            1  Lung Cancer Diagnosed
99        1           2            0         No Lung Cancer

[100 rows x 4 columns]
```

VI. ADVANTAGES AND DRAWBACKS

The rule-based and deep learning combined systems had multiple advantages according to the experimentation. It provided a filtering system to filter out possible cases that were too difficult or mixed for the rule-based system to evaluate. By handling these simpler cases this saves the deep learning system time and allows for more efficiency when handling the more complex cases. The deep learning system using neural networks was better suited for the more complex cases because of its ability to capture patterns within the dataset. This was possibly more of an efficient that allowed for more focus on the complicated cases, rather than just a simple rule-based expert system handling them. Neural networks proved to be a positive system for medical diagnosis using the datasets without the use of CNNs for medical imaging.

The drawbacks were that without medical imaging the accuracy was not as high as it could have been. With medical imaging alongside diagnosis through the symptoms the accuracy could possibly have been higher. This was a major drawback, however, using neural networks in this research was still useful to show the accuracy alongside the rule-based expert system. Neural networks have an issue regarding bias. The bias is set to help with the predictions. However, with a dataset that has uneven labels, such as the one this system used, there is often a loss in accuracy because of it. The neural network attempts to find patterns within the data and due to the guessing “True” to lung cancer yielded a higher accuracy the system had an innate bias. Which in turn made the system not pay attention to the false classes. However, this was mostly fixed by a class weight system that created more weight on the “False” cases, but not entirely. There is also reason to believe that the rule-based system to wrongly classifies the cases as complex or as simple. This may lower the accuracy of the overall system by sending simpler cases to the deep learning model and attempting to solve the more complex ones. There is also the matter of scalability, as the rule-based system has to be changed manually for predicting on other datasets.

VII. SUMMARY AND CONCLUSION

In summary, AI-based methods for medical diagnostic have seen much development. Rule-based expert systems have been studied specifically for this matter due to their ability to give accurate diagnostic using the prior knowledge of professionals. Deep learning has also played a huge role in the field for its abilities to learn and find patterns within various data as well as imaging. The hybrid method of using both systems offers a unique perspective on what the possibilities are for detection accuracy. Despite limitations, the findings did show that both systems can work great and have a high accuracy score. The rule-based system was effective in its predictions and the neural network did a respectable job with its ability to predict lung cancer cases. Research into methods such as deep learning using medical imaging showed promise during the research. Combining both a refined rule-based system alongside deep learning for medical imaging would rule to have far higher accuracy besides symptom testing. Future research incorporating deep learning for medical imaging alongside the rule-based system, could refine the results and support the growing research in the technology for the medical field.

VIII. REFERENCES

[1] S. S. Gulavani and R. V. Kulkarni, “A Review of Knowledge Based Systems In Medical Diagnosis” in International Journal of Information Technology and Knowledge Management, vol. 2., no. 2, July-December 2009, pp. 269-275.

[2] D. Painuli, D. Mishra, and Nirvikar, “Rule Based Expert System for Medical Diagnosis-A Review” in International Journal of Engineering Technology, Management and Applied Sciences, vol 4., issue 12, December 2016.

[3] K. Cincar and T. Ivașcu, "Rule-based Health Status Evaluation System," in *2021 International Conference on e-Health and Bioengineering (EHB)*, Iasi, Romania, 2021, pp. 1-4, doi: 10.1109/EHB52898.2021.9657694.

[4] M.W. Kattan, “Expert Systems in Medicine” International Encyclopedia of the Social & Behavioral Sciences, Pergamon, 2001, pp. 5135-5139, doi: 10.1016/B0-08-043076-7/00556-8.

[5] MedlinePlus [Internet]. Bethesda (MD): National Library of Medicine (US); [updated]. Lung cancer; [updated July 29, 2024]; [about 5 p.]. Available from: https://medlineplus.gov/lungcancer.html#cat_92.

[6]Yeung, H. Co, A. T. H. Mulgirigama, S. Johnston, S. Geller, and A. E. Khandwala, "Artificial intelligence in clinical decision support and screening: a review," npj Digital Medicine, vol. 4, no. 1, p. 146, 2021. [Online]. https://www.nature.com/articles/s41746-021-00438-z

[7] A. Biswal, “Top 10 Deep Learning Algorithms You Should Know in (2021),” Simplilearn.com, Feb. 16, 2023. https://www.simplilearn.com/tutorials/deep-learning-tutorial/deep-learning-algorithm

[8] K. Dhawan and U. Agrawal, “Training and testing a CNN-based engine for brain MRI scan classification and segmentation,” medrxiv, Jul. 06, 2023. https://www.medrxiv.org/content/10.1101/2023.07.03.23292122v1

[9] J. B. Awotunde, R. Panigrahi, B. Khandelwal, A. Garg, and A. K. Bhoi, “Breast cancer diagnosis based on hybrid rule-based feature selection with deep learning algorithm,” Research on Biomedical Engineering, Jan. 2023, doi: https://doi.org/10.1007/s42600-022-00255-7.

[10] M. Bakator and D. Radosav, “Deep Learning and Medical Diagnosis: A Review of Literature,” Multimodal Technologies and Interaction, vol. 2, no. 3, p. 47, Aug. 2018, doi: https://doi.org/10.3390/mti2030047.

[11] Varin Senthil, “Deep Learning and Rule-Based Hybrid Approach to Improve the Accuracy of Early Detection of Skin Cancer,” Journal of Clinical & Experimental Dermatology Research, vol. 14, no. 5, pp. 1–4, Sep. 2023, doi: https://doi.org/10.35248/2168-9296.23.12.446.