CT Scan Image Classification Using SIFT and SVM

Animesh Uttekar (115382330)

Shashi Varun Kalimera (115362761)

1. INTRODUCTION

Medical imaging stands at the forefront of modern healthcare, playing a pivotal role in the diagnosis and treatment of various medical conditions. As technological advancements propel medical imaging into new realms, the escalating volume of imaging data necessitates innovative approaches for efficient analysis and interpretation. This project addresses the imperative need for automated tools in medical imaging, specifically focusing on machine learning techniques to revolutionize the classification of different body parts in CT scan images. The overarching goal is to streamline the image sorting process, ultimately enhancing the efficiency of diagnostic procedures.

1.1 Background

Traditionally, the process of categorizing CT scans into specific body parts involves manual intervention, where a healthcare professional visually inspects each image and assigns it to the relevant anatomical region. This manual sorting can be time-consuming and prone to human error. As medical imaging technology continues to advance, the sheer volume of imaging data generated necessitates more efficient and accurate methods for image categorization.

The motivation behind this project stems from the desire to harness the power of machine learning to automate the classification of CT scans, particularly focusing on the body parts such as chest, brain, and abdomen. Automation of this process can significantly reduce the workload on healthcare professionals, enabling them to allocate more time to critical tasks and enhancing the overall efficiency of medical imaging facilities.

1.2 Goals

The primary goal of this project is to develop a robust and accurate system for the automated classification of CT scans into distinct body parts. This involves the extraction of relevant features from the images, followed by the application of machine learning techniques to classify the images into predefined categories. The specific objectives include:

Automated Classification: Create a reliable system to automatically categorize CT scans into distinct body parts.

Feature Extraction Excellence: Utilize advanced techniques, including HOG and SIFT, to extract pertinent features capturing unique patterns in CT scan images.

Precision with Machine Learning: Implement an SVM model for accurate classification of CT scans into predefined categories such as chest, brain, and abdomen.

Seamless Automation: Develop a system seamlessly integrating into existing medical imaging workflows, streamlining the sorting process for enhanced diagnostic efficiency.

Adaptability and Expansion: Ensure the system's flexibility to diverse medical imaging datasets by enabling straightforward incorporation of additional anatomical region.

2. Data Preparation and Feature Extraction and classification

2.1 Dataset:

The dataset is stored in the Google Drive directory -

/content/drive/MyDrive/CT_Scan_images_2.

It contains CT scan images categorized into 'abdomen,' 'brain,' and 'chest.'

Brain CT scan images are taken from OASIS dataset - https://www.oasis-brains.org/

Abdomen and Chest CT scan images are taken from TCIA dataset – https://cancerimagingarchive.net/

2.2 Data Preparation:

- The medical images are preprocessed by converting them to "JPEG" image format and resizing. This ensures a consistent input size and simplifies feature extraction.
- Standardizing image size is crucial for consistent feature extraction.

2.3 SIFT Feature Extraction in CT Scan Images:

This project involves utilizing the SIFT (Scale-Invariant Feature Transform) algorithm for feature extraction from CT scan images, followed by classification using the SVM (Support Vector Machine) algorithm.

SIFT is a computer vision algorithm used to detect and describe local features in images. In this project, it's applied to CT scan images for identifying keypoints and extracting their descriptors.

2.3.1 Keypoint Detection:

The SIFT algorithm detects keypoints in CT scan images. These keypoints are specific areas in the image that the algorithm finds significant.

For each image, resized to 128×128 pixels, SIFT computes keypoints and their descriptors.

The keypoint descriptor is a 128-dimensional vector representing the local gradient information around the keypoint.

2.3.2 Descriptor Aggregation:

All descriptors from the CT scan images are collected. This aggregated data forms the basis for creating a visual vocabulary.

2.3.3 Vocabulary Creation Using Mini-Batch K-Means:

The descriptors are clustered using the Mini-Batch K-Means algorithm to form a vocabulary of size 200.

Mini-Batch K-Means, a variant of K-Means, optimizes the clustering process by processing small random batches of data, making it efficient for large datasets.

The clustering process minimizes the within-cluster sum of squares:

$\sum_{i=1}^{k}\sum_{x \in S_i}||x-\mu_i||^2$, where S_i is the set of points in the i-th cluster and μ_i is the centroid of S_i.

2.3.4 Histogram of Features:

Each image's descriptors are mapped to the closest cluster centroid in the vocabulary, and a histogram is created to represent the frequency of each cluster's occurrence in the image.

The histograms are normalized to have unit length, which helps in reducing the effects of varying illumination and contrast in different CT scan images.

2.4 SVM for Classifying CT Scan Images:

SVM is a supervised learning algorithm used for classification or regression challenges. In this project, it classifies CT scan images based on their extracted SIFT features.

2.4.1 Training the Classifier:

The SVM classifier is trained with the SIFT features (histograms) of the CT scan images. We have used a linear kernel, but other kernels like RBF or polynomials can also be employed.

The linear SVM aims to find the optimal separating hyperplane

$w \cdot x + b = 0$, where w is the weight vector and b is the bias.

The optimization problem for a linear SVM is to minimize $\frac{1}{2}||w||^2$ subject to **$y_i(w.x_i+b)>=1$** for each data point(x_i,y_i), where y_i are the labels(classifications).

2.4.2 Prediction and Evaluation:

The trained SVM model is used to classify test images based on their SIFT features.

The performance of the model is evaluated using accuracy, precision, recall, and a confusion matrix. The confusion matrix is particularly useful in understanding the classification performance for each class.

3. Algorithm and Implementation

3.1 SIFT Feature Extraction and Visual Vocabulary Creation:

3.1.1 Importing Libraries and Mounting Google Drive:

Necessary libraries and modules for processing includes OpenCV, array operations (NumPy), file and directory handling (os), clustering (MiniBatchKMeans), image manipulation (PIL), data analysis (Pandas), and distance computation (scipy). Google Drive is also mounted to access the dataset.

```
import cv2

import numpy as np

import os

from sklearn.cluster import MiniBatchKMeans
```

```
from PIL import Image

import pandas as pd

from scipy.spatial.distance import cdist

from google.colab import drive

# Mounting Google Drive

drive.mount('/content/drive')
```

3.1.2 Setting up Dataset Directory and Image Size:

The dataset directory and the desired image size for resizing are specified.

```
# Directory containing the dataset

dataset_dir = "/content/drive/MyDrive/CT_Scan_images_2"   # path
to the dataset

# Resizing images to this size

image_size = (128, 128)
```

3.1.3 SIFT Feature Extractor Initialization:

The Scale-Invariant Feature Transform (SIFT) algorithm is initialized using the cv2.SIFT_create() method. SIFT is chosen for its robustness to scaling, rotation, and illumination changes, making it suitable for extracting distinctive features from CT scan images.

```
sift = cv2.SIFT_create()
```

3.1.4 Collecting Descriptors for Vocabulary:

This loop iterates through the dataset, reads each CT scan image, converts it to grayscale, resizes it, and applies the SIFT algorithm to detect and compute descriptors. These descriptors are then accumulated in the all_descriptors list.

```
all_descriptors = []

for subdir, dirs, files in os.walk(dataset_dir):
```

```
    for file in files:

        img_path = os.path.join(subdir, file)

        pil_img = Image.open(img_path).convert('L')

        img = np.array(pil_img)

        img_resized = cv2.resize(img, image_size)

        _, descriptors = sift.detectAndCompute(img_resized, None)

        if descriptors is not None:

            all_descriptors.extend(descriptors)
```

3.1.5 Clustering Descriptors to Create Visual Vocabulary:

Mini-Batch K-Means clustering is applied to the collected descriptors. This clustering process groups similar descriptors into clusters, forming a visual vocabulary. The resulting centroids (vocabularies) are saved in 'vocab.npy'.

```
vocab_size = 200

minibatch_kmeans = MiniBatchKMeans(n_clusters=vocab_size,
random_state=0, batch_size=300)

minibatch_kmeans.fit(np.array(all_descriptors))

vocab = minibatch_kmeans.cluster_centers_

np.save('vocab.npy', vocab)
```

3.1.6 Feature Extraction using Visual Vocabulary:

Feature extraction is executed using the created visual vocabulary. It iterates through the dataset, calculating the distance of each descriptor to the vocabulary, assigning the descriptor to a cluster, and generating a histogram of visual word occurrences for each image. These histograms are normalized and stored in features_list along with corresponding labels in labels_list.

```
features_list = []

labels_list = []
```

```python
label_mapping = {'abdomen': 0, 'brain': 1, 'chest': 2}

for subdir, dirs, files in os.walk(dataset_dir):

    for file in files:

        img_path = os.path.join(subdir, file)

        label_name = os.path.basename(subdir).lower()

        if label_name not in label_mapping:

          continue

        label = label_mapping[label_name]

        pil_img = Image.open(img_path).convert('L')

        img = np.array(pil_img)

        img_resized = cv2.resize(img, image_size)

        _, descriptors = sift.detectAndCompute(img_resized, None)

        if descriptors is not None:

            dist = cdist(descriptors, vocab, 'euclidean')

            bin_assignment = np.argmin(dist, axis=1)

            image_hist = np.zeros(vocab_size)

            for bin_id in bin_assignment:

                image_hist[bin_id] += 1

            image_hist /= np.linalg.norm(image_hist)

            features_list.append(image_hist)

            labels_list.append(label)
```

3.1.7 Convert to DataFrame and Save to CSV:

The extracted features and corresponding labels are then converted to Pandas DataFrames (features_df and labels_df). These DataFrames are saved as CSV files (***'features.csv' and 'labels.csv'***, respectively) for later use in the machine learning classification phase. The index=False parameter ensures that the CSV files are saved without row indices.

```
# Convert to DataFrame and save to CSV

features_df = pd.DataFrame(features_list)

labels_df = pd.DataFrame(labels_list, columns=['label'])

features_df.to_csv('features.csv', index=False)

labels_df.to_csv('labels.csv', index=False)
```

SIFT descriptors are normalized into histograms, reducing sensitivity to variations in image sizes and scales.

3.2 SVM Classification and Evaluation:

3.2.1 Load Features and Labels:

The features and corresponding labels are loaded from the CSV files ('features.csv' and 'labels.csv').

```
features_df = pd.read_csv('features.csv')

labels_df = pd.read_csv('labels.csv')
```

3.2.2 Split Data into Training and Testing Sets:

The data is split into training and testing sets using the train_test_split function from scikit-learn. The test_size parameter defines the proportion of the dataset allocated for testing. The dataset is split into training and testing sets (*80-20 split*).

```
# Split data into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(features_df,
labels_df['label'], test_size=0.2, random_state=42)
```

3.2.3 Initialize and Train the SVM Classifier:

A Support Vector Machine (SVM) classifier is initialized with a *linear kernel*. Other kernel options like 'rbf' (Radial basis function) and 'poly' (Polynomial) can be experimented with. The classifier is then trained on the training set**.**

```
# Initialize and train the SVM classifier

svm_classifier = SVC(kernel='linear')  # we can use different kernels like 'rbf', 'poly', etc.

svm_classifier.fit(X_train, y_train)
```

3.2.4 Predict on the Test Set:

The trained SVM classifier is used to predict labels for the test set.

```
# Predict on the test set

y_pred = svm_classifier.predict(X_test)
```

3.2.5 Evaluate the Classifier:

The classification report, accuracy score, and other metrics are printed to evaluate the performance of the SVM classifier on the test set.

```
# Evaluate the classifier

print("Classification Report:")

print(classification_report(y_test, y_pred))

print("Accuracy Score:", accuracy_score(y_test, y_pred))
```

3.2.6 Generate and Display the Confusion Matrix:

The confusion matrix is computed using scikit-learn's confusion_matrix function and then visualized using a heatmap with seaborn and Matplotlib.

```
cm = confusion_matrix(y_test, y_pred)

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')

plt.title('Confusion Matrix')

plt.xlabel('Predicted Label')

plt.ylabel('True Label')

plt.show()
```

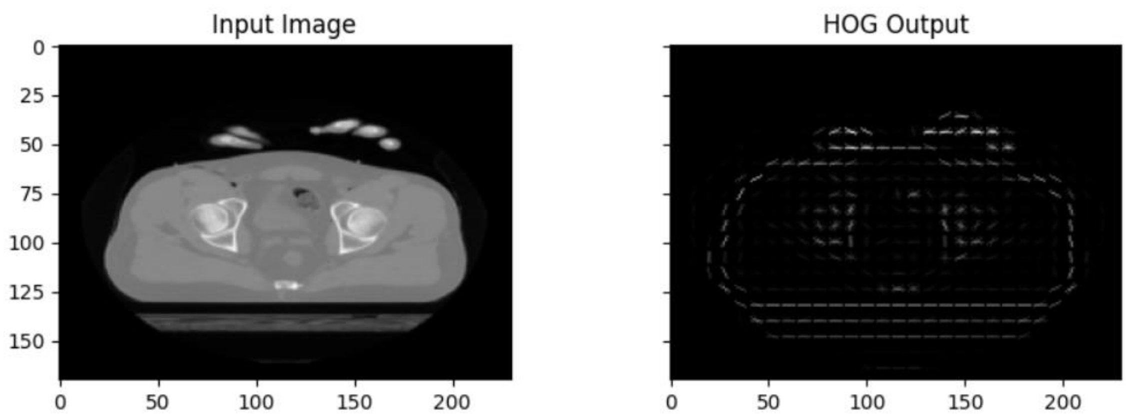
4. Comparison with other Techniques:

4.1 Why SIFT Over Other Feature Extraction Methods?

4.1.1 Initial Approach with HOG:

- HOG (Histogram of Oriented Gradients) was initially considered for its efficacy in edge and gradient structure detection, valuable in image-based machine learning tasks.
- The application of HOG to CT scan images created a very high-dimensional feature space, with over 80,000 features, which is computationally inefficient and challenging for classifiers to process.

```
Saved HOG features to CSV file. Number of features extracted per image: 81162
```



4.1.2 Challenges with High Dimensionality:

- Managing such a large number of features increases computational demands and training time significantly.
- High-dimensional spaces risk overfitting, where the classifier may learn noise as opposed to genuine patterns.
- While dimensionality reduction through PCA (Principal Component Analysis) is possible, it might oversimplify the data, resulting in the loss of crucial information.

4.1.3 Advantages with SIFT:

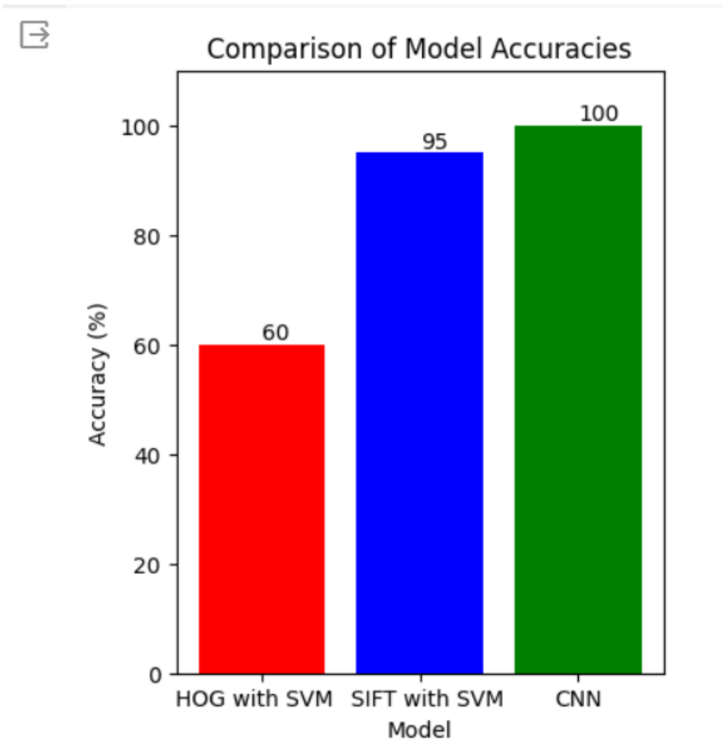
- SIFT features are invariant to image scaling and rotation, and partially invariant to changes in illumination and 3D perspective, making them more robust for medical image analysis.

- SIFT extracts informative keypoints that are distinct and descriptive, aligning with the specificity required in medical image classification.

4.1.4 Optimization with SIFT:

- By constraining the SIFT vocabulary size to 200, we reduced the feature space to a manageable scale.
- PCA was then applied to these SIFT features, refining them to the most informative 200, thus balancing detail with computational simplicity.

4.2 Comparison Bar graph:



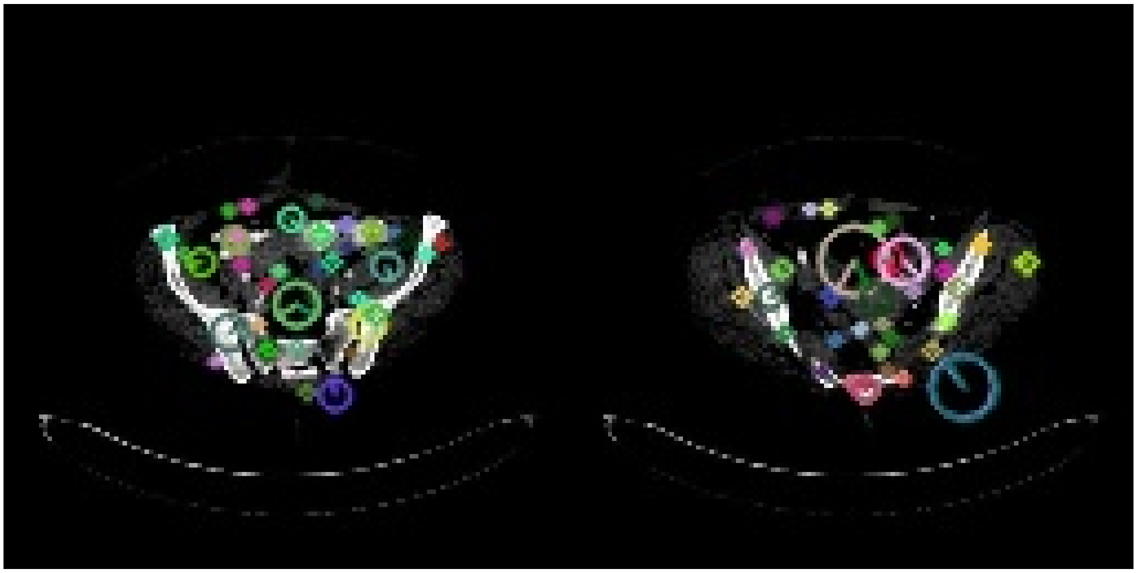
4.3 CNN Implementation:

The Convolutional Neural Network (CNN) developed using TensorFlow and Keras demonstrated exceptional performance, achieving nearly *100% accuracy rate* in classifying body parts from CT scans. The model's depth and feature learning capabilities were key contributors to this success. While its automated feature extraction streamlined the process, the CNN's black-box nature raises interpretability concerns, crucial in clinical contexts. Overall, this implementation suggests that CNNs hold significant promise for automating medical image classification tasks, potentially enhancing diagnostic precision.

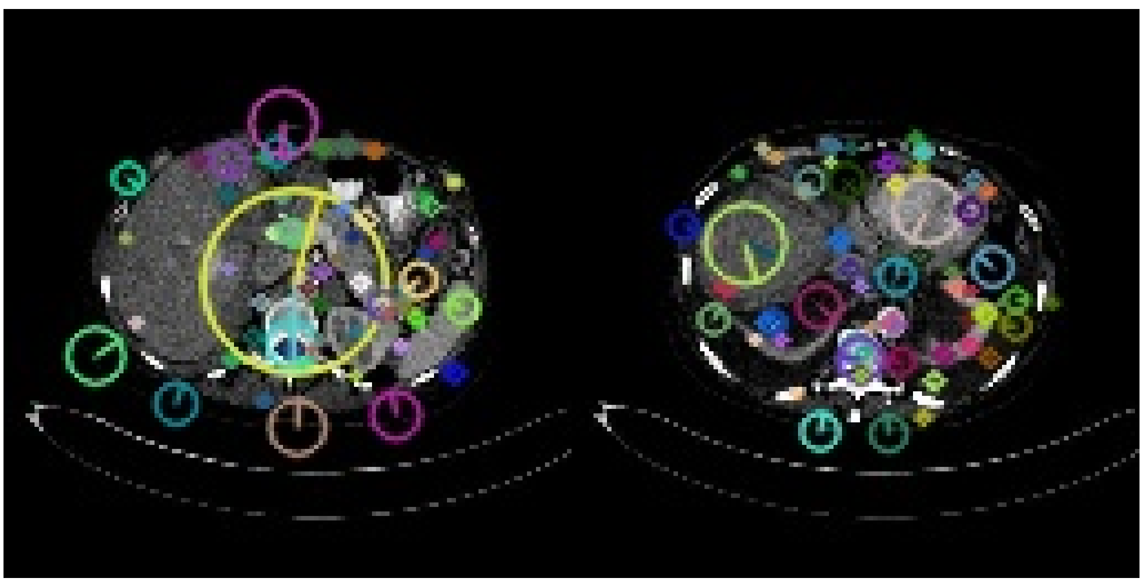
5. Results and Conclusions:

5.1 Results:

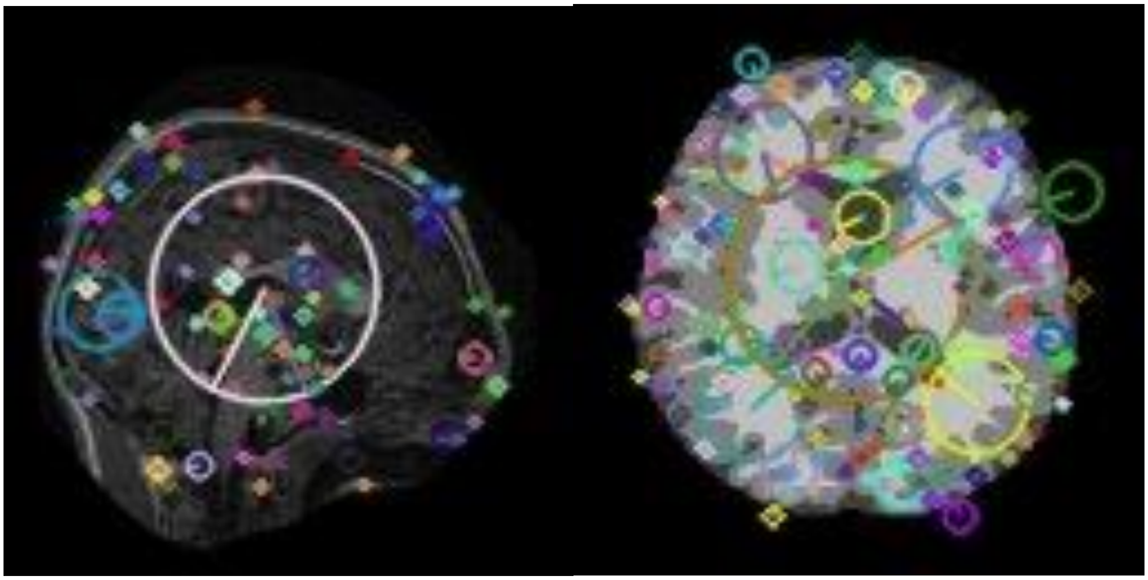
5.1.1 SIFT Feature keypoints of Abdomen CT scan images:



5.1.2 SIFT Feature keypoints of Chest CT scan images:



5.1.3 SIFT Feature keypoints of Brain CT scan images:



5.2 Understanding Results:

The colored circles in the above feature extracted CT scan images represent keypoints identified by the SIFT algorithm. These keypoints are distinctive areas in the image that SIFT has recognized as significant. The colors and sizes of the circles may correspond to the scale at which the feature was detected and the orientation of the keypoint in the local area of the image. The presence of multiple keypoints across different regions of the abdomen, chest and brain scan signifies a rich feature set, which could be crucial for diagnosing various conditions.

5.2.1 Significance of Keypoint Colors and Sizes:

The colors and sizes of the keypoints serve more than an aesthetic purpose; they provide information about the feature's characteristics. The size of a keypoint circle usually represents the scale, with larger circles indicating features that are identifiable at larger scales (or lower resolutions), and smaller circles representing features that are visible at finer scales (or higher resolutions). The color coding, while not standard, could be used to differentiate between different orientations or other feature properties defined during the SIFT extraction process

5.3 Classification report:

F1-Score:

The F1-score is the weighted average of precision and recall, to show that a classifier has a good value for both recall and precision.

F1-score for class 0: 0.94 (94%)

F1-score for class 1: 1.00 (100%)

F1-score for class 2: 0.93 (93%)

*Overall Accuracy: 0.95 (**95%**)*

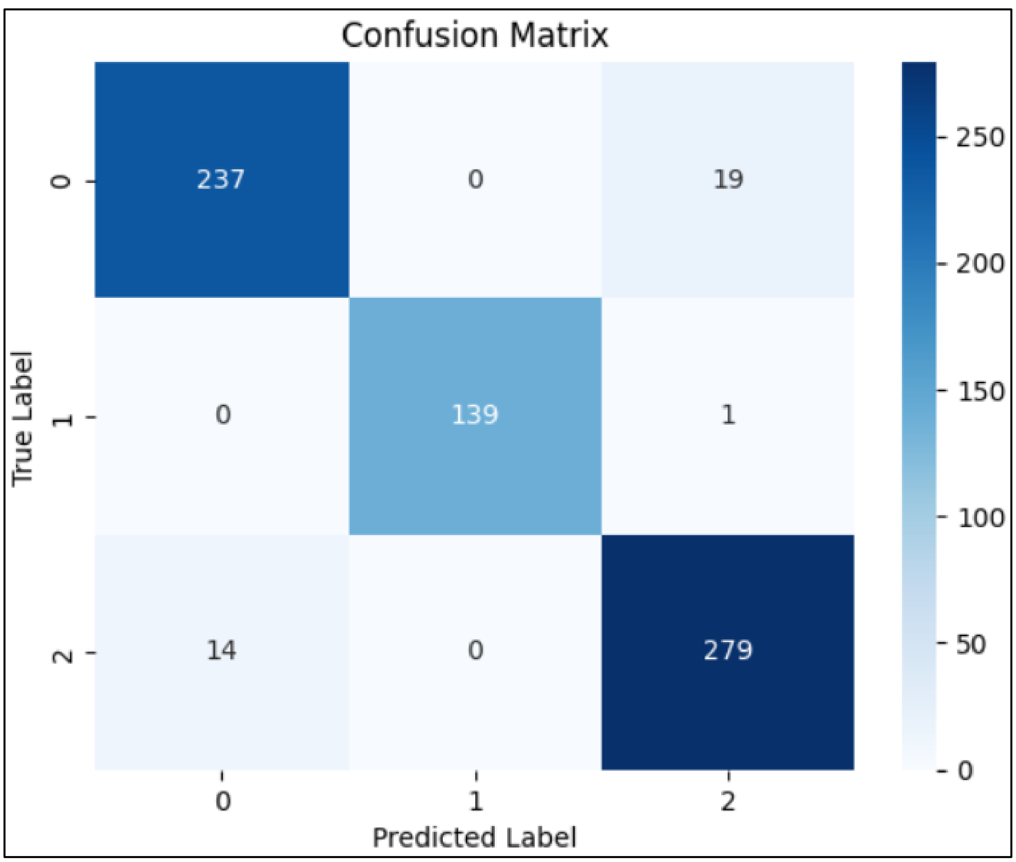
```
Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.93      0.93       256
           1       1.00      0.99      1.00       140
           2       0.93      0.95      0.94       293

    accuracy                           0.95       689
   macro avg       0.96      0.96      0.96       689
weighted avg       0.95      0.95      0.95       689

Accuracy Score: 0.9506531204644412
```

5.4 Confusion Matrix:



For class 0, there are 237 true positives, 0 false positives, and 19 false negatives.
For class 1, there are 139 true positives, 0 false positives, and 1 false negative.
For class 2, there are 279 true positives, 14 false positives, and 0 false negatives.

5.5 Comparison with Theory:

The theory behind SVMs and SIFT features is rooted in machine learning and computer vision literature. The project aligns with theoretical expectations for image classification using SVM and SIFT features.

- Bag-of-Visual-Words Model: The code aligns with the bag-of-visual-words model theory, where images are represented as histograms of visual words.
- SVM for Image Classification: The use of an SVM classifier aligns with theoretical foundations for image classification tasks.

5.6 Summary of Findings:

The classifier seems to perform well with high precision, recall, and F1-score for all classes. The overall accuracy is 95%, indicating that the model is effective in making correct predictions. The confusion matrix provides more detailed information on the types of errors made by the classifier. There are some misclassifications in class 0 (abdomen) and class 2 (chest) due to similarity in few abdomens and chest CT scan images, but overall, the classifier performs well across all classes.

5.7 CONCLUSION:

The project successfully classifies medical images into 3 categories.

- The classification report provides insights into precision, recall, and F1-score for each class.
- The accuracy score gives an overall performance measure which is 95%.
- The confusion matrix provides additional insights into the classifier's performance, revealing that the model has a low number of false positives and false negatives. While there are some misclassifications in classes 0 and 2, these errors are relatively small compared to the total number of instances, and the overall performance remains robust.
- Overall, the model serves as a solid foundation for classification tasks on the dataset.

6. References and Citations:

- https://liverungrow.medium.com/sift-bag-of-features-svm-for-classification-b5f775d8e55f
- https://www.oasis-brains.org/

- https://cancerimagingarchive.net/
- https://www.researchgate.net/publication/358583456_Feature_Extraction_Methods_for_CT-Scan_Images_Using_Image_Processing
- https://medium.com/analytics-vidhya/image-classification-using-machine-learning-support-vector-machine-svm-dc7a0ec92e01
- https://medium.com/@deepanshut041/introduction-to-sift-scale-invariant-feature-transform-65d7f3a72d40