Project Report for Digital Image Processing (Phase B)

Maria Eleni Chochlidaki
Michail Samaritakis

Outline:

- Project's Goal
- Project's Files
- Project's Progress
    1. Imports and Initial Setup
    2. Parsing XML Annotations
    3. Processing Command-Line Arguments
    4. Image Preprocessing and Stitch Counting
    5. Feature Extraction and Machine Learning
    6. Writing Results
    7. Main Execution
    8. Output Interpretation
    9. Conclusion
- Summary
- How to Run the Project

Project's Goal:

The purpose of this project is to detect the quantity of stitches and incisions from a variety of images produced during a surgical operation. This can be accomplished by using computer vision and digital image processing techniques.

Project's Files:

- resources/incision_couples: Contains the given images.

- resources/test_images: Contains images that will be used as arguments for the testing of the project.

- resources/annotations.json: Contains parsed annotation data in JSON format. This is used to map the incision and stitch points for each image.

- resources/annotations.xml: Original XML file containing annotation data before conversion to JSON. This XML is parsed and converted into a more accessible format.

- src/convert_xml_to_json.py: A script for converting the XML annotation file to a JSON format for easier processing. This ensures that the data is structured and accessible for subsequent analysis.

- src/run.py: The main script that runs the entire process: from reading annotations, preprocessing images, extracting features, training a classifier, and outputting results.

- data.json: Contains sample data for testing or as a template. This includes filenames, incision polylines, crossing positions, and angles.

- root/report_images: Contains the report_images folder which has the images that have been used as arguments without including those with bad data.

Project's Progress:

Here's a step-by-step explanation of what each segment of the code does and what the output represents. The code includes specific comments for each action and method:

1. Imports and Initial Setup

The project begins with necessary imports from various libraries:

- Deep Learning: `tensorflow.keras.layers`, `tensorflow.keras.models`

- Machine Learning and Data Processing: `sklearn.model_selection`, `sklearn.preprocessing`, `imblearn.over_sampling`, `sklearn.metrics`

- Image Processing: `skimage.filters`, `skimage.segmentation`, `cv2`, `matplotlib.pyplot`

- General Purpose: `numpy`, `xmltodict`, `argparse`, `json`, `csv`, `sys`, `os`, `math`

The initial setup includes defining the XML path for annotations.

2. Parsing XML Annotations

The script reads an XML file containing image annotations using `xmltodict`. The parsed document contains metadata and image information, which is used later for processing. This setup extracts essential keys and specific fields for further analysis.

3. Processing Command-Line Arguments

The function `parse_arguments` handles command-line arguments using `argparse`. It defines:

- Output CSV file

- Visual mode flag

- List of image filenames

- Method for counting stitches (`hough` or `predict`)

4. Image Preprocessing and Stitch Counting

Image Preprocessing:

- Converts images to grayscale.

- Applies Gaussian blur to smooth the images.

Stitch Counting:

- Uses Canny edge detection and Hough Transform for line detection in images to count stitches.

The `evaluate_image_quality` function assesses the image quality based on blur, brightness, and presence of lines using Hough Transform.

5. Feature Extraction and Machine Learning

Feature Extraction:

- Extracts features from annotations, including segment lengths and angles.

- Aggregates features to create uniform feature vectors.

Machine Learning:

- Standardises features using `StandardScaler`.

- Handles class imbalance with `RandomOverSampler`.

- Splits data into training and testing sets.

- Uses `GridSearchCV` to find the best hyperparameters for an SVM classifier.

- Trains and evaluates the classifier, printing confusion matrix and classification report.

6. Writing Results

The `write_to_csv` function writes the results of the image processing (image filenames and stitch counts) to a specified CSV file.

7. Main Execution

In the main execution block:

- Arguments are parsed.

- Annotations are loaded.

- Features and labels are extracted from the annotations.

- A classifier is trained if sufficient class diversity exists.

- For each image, the script:

  - Evaluates image quality.
  - Processes the image using the specified method (`hough` or `predict`).
  - Writes results to the output CSV file.

8. Output Interpretation

-The results of the project are stored in a CSV file containing the filenames and corresponding stitch counts. The project provides detailed feedback on each image processed, including errors(number of stitches = 1) and the number of stitches detected.

Results into output.csv file using mode ‘predict’:

```
filename,n_stitches
360_F_64771693_ncondhOJwNdvLjBfeIwswLqhsavUSSY5.jpg,-1
istockphoto-186989173-612x612.jpg,134
SA_20220620-102836_e8o0aggtd2us_incision_crop_0_start.jpg,-1
SA_20220620-102836_e8o0aggtd2us_incision_crop_0.jpg,134
SA_20220620-103348_3imoxskpwsvo_incision_crop_0_start.jpg,134
SA_20220620-104018_5w0i85t736jm_incision_crop_0_start.jpg,134
SA_20220620-105545_tlmnkcr9sjlm_incision_crop_0_start.jpg,134
SA_20220620-105330_tc2abmwmh9mt_incision_crop_0.jpg,134
SA_20220707-190320_tl4ev95290pp_incision_crop_0_start.jpg,-1
SA_20220707-185800_qz8chub67fgk_incision_crop_0_start.jpg,134
```

Results into output.csv file using mode ‘hough’:

```
filename,n_stitches
360_F_64771693_ncondhOJwNdvLjBfeIwswLqhsavUSSY5.jpg,-1
istockphoto-186989173-612x612.jpg,6
SA_20220620-102836_e8o0aggtd2us_incision_crop_0_start.jpg,-1
SA_20220620-102836_e8o0aggtd2us_incision_crop_0.jpg,3
SA_20220620-103348_3imoxskpwsvo_incision_crop_0_start.jpg,2
SA_20220620-104018_5w0i85t736jm_incision_crop_0_start.jpg,1
SA_20220620-105545_tlmnkcr9sjlm_incision_crop_0_start.jpg,0
SA_20220620-105330_tc2abmwmh9mt_incision_crop_0.jpg,2
SA_20220707-190320_tl4ev95290pp_incision_crop_0_start.jpg,-1
SA_20220707-185800_qz8chub67fgk_incision_crop_0_start.jpg,0
```

Bad data:

```
Error: Image 360_F_64771693_ncondhOJwNdvLjBfeIwswLqhsavUSSY5.jpg could not be processed
Error: Image SA_20220620-102836_e8o0aggtd2us_incision_crop_0_start.jpg could not be processed
Error: Image SA_20220707-190320_tl4ev95290pp_incision_crop_0_start.jpg could not be processed
```

- The number of detected stitches in annotations is printed for each image.

```
Number of stitches from annotations:
resources/incision_couples/SA_20220620-102621_8ka1kmwpywxv_incision_crop_0.jpg , 2 # image contains 2 stiches
resources/incision_couples/SA_20220620-102621_8ka1kmwpywxv_incision_crop_0_start.jpg , 1 # image contains 1 stiches
resources/incision_couples/SA_20220620-102836_e8o0aggtd2us_incision_crop_0.jpg , 2 # image contains 2 stiches
resources/incision_couples/SA_20220620-102836_e8o0aggtd2us_incision_crop_0_start.jpg , -1 # image could not be processed
resources/incision_couples/SA_20220620-103036_3vhpgd00n9vt_incision_crop_0.jpg , 4 # image contains 4 stiches
resources/incision_couples/SA_20220620-103036_3vhpgd00n9vt_incision_crop_0_start.jpg , 3 # image contains 3 stiches
resources/incision_couples/SA_20220620-103348_3imoxskpwsyo_incision_crop_0.jpg , 3 # image contains 3 stiches
```

- Analysis of the Confusion Matrix and Classification Report:

```
Confusion Matrix:
[[6 0 0 0 0 0 0]
 [0 7 0 0 0 0 0]
 [0 0 5 0 0 0 0]
 [0 1 3 1 0 0 0]
 [0 1 1 1 3 0 0]
 [0 0 2 0 1 6 0]
 [0 0 0 0 0 0 6]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         6
           1       0.78      1.00      0.88         7
           2       0.45      1.00      0.62         5
           3       0.50      0.20      0.29         5
           4       0.75      0.50      0.60         6
           5       1.00      0.67      0.80         9
           6       1.00      1.00      1.00         6

    accuracy                           0.77        44
   macro avg       0.78      0.77      0.74        44
weighted avg       0.81      0.77      0.76        44
```

Confusion Matrix:

The confusion matrix displays the performance of the classification model across different classes (0 through 6). Each row represents the actual class, while each column represents the predicted class.

[[6 0 0 0 0 0 0] # Class 0: All 6 instances correctly classified

[0 7 0 0 0 0 0] # Class 1: All 7 instances correctly classified

[0 1 4 0 0 0 0] # Class 2: 1 instance misclassified as Class 1, 4 correctly classified

[0 1 1 3 0 0 0] # Class 3: 1 instance misclassified as Class 1, 1 as Class 2, 3 correctly classified

[0 2 3 1 0 0 0] # Class 4: 2 misclassified as Class 1, 3 as Class 2, 1 as Class 3

[0 0 2 1 6 0 0] # Class 5: 2 misclassified as Class 2, 1 as Class 3, 6 correctly classified

[0 0 0 0 0 0 6]] # Class 6: All 6 instances correctly classified

Classification Report:

The classification report provides detailed metrics for each class: precision, recall, f1-score, and support.

- Precision: The ratio of true positives to the sum of true and false positives. Precision indicates the accuracy of the positive predictions.

- Recall: The ratio of true positives to the sum of true positives and false negatives. Recall indicates the ability to find all relevant instances.

- F1-Score: The harmonic mean of precision and recall. It provides a balance between the two metrics.

- Support: The number of actual instances in the dataset for each class.

	precision	recall	f1-score	support
0	1.00	1.00	1.00	6
1	0.70	1.00	0.82	7
2	0.50	0.20	0.29	5
3	0.25	0.60	0.35	5
4	0.50	0.17	0.25	6
5	1.00	0.67	0.80	9
6	1.00	1.00	1.00	6
accuracy			0.68	44
macro avg	0.71	0.66	0.64	44
weighted avg	0.74	0.68	0.67	44

Analysis of Metrics by Class:

- Class 0 and Class 6:

Both classes have perfect precision, recall, and f1-score, indicating that the model performs very well in identifying these classes correctly.

- Class 1:

High precision (0.70) and perfect recall (1.00) suggest the model is good at identifying instances of Class 1, though there are some false positives.

- Class 2:

Lower precision (0.50) and recall (0.20) indicate that the model struggles with correctly identifying Class 2, leading to a low f1-score (0.29).

- Class 3:

Low precision (0.25) but higher recall (0.60), suggesting the model identifies many instances of Class 3 but also has a significant number of false positives.

- Class 4:

Precision (0.50) and recall (0.17) are low, indicating poor performance in identifying Class 4 instances.

- Class 5:

High precision (1.00) but moderate recall (0.67), showing the model correctly identifies some instances but misses others.

Overall Metrics:

- Accuracy: 0.68

The overall proportion of correct predictions.

- Macro Average:

- Precision: 0.71
- Recall: 0.66
- F1-Score: 0.64

- Weighted Average:

- Precision: 0.74
- Recall: 0.68

- F1-Score: 0.67

Conclusion regarding the output:

The model performs very well on certain classes (0, 1, and 6) but struggles with others (2, 3, and 4). The overall accuracy of 68% suggests that there is room for improvement, particularly in handling class imbalance and better identifying difficult classes. Improving precision and recall for underperforming classes could significantly enhance the model's overall performance.

9. Conclusion

The project successfully integrates multiple techniques for image processing, feature extraction, and machine learning to count stitches in surgical images. The modular approach ensures flexibility in methods and robustness in handling different types of input data.

Summary:

This project combines image processing and machine learning to analyse surgical images for stitch counting. It preprocesses images, extracts relevant features, and trains a classifier to predict stitch counts, with results stored in a CSV file. The project does not provide completely correct results but it works properly with detecting bad data in an image. Although, the results produced by Hough Transform(mode ‘hough’) have a smaller error rate compared to using the ‘predict’ mode.

How to run the Project:

We created two methods for the user to select which type of project’s execution prefers.

Argument Parsing:
Added a *-m* or *--method* argument to specify the method for counting stitches. The choices are ‘*hough’* and *‘predict’*, with ‘*hough*’ as the default.

Main Execution:
Depending on the user's choice, the script either uses process_image_with_hough_transform or classifier.predict(features) to count the stitches.

Visual Mode:
In visual mode, displays the result using the chosen method.

Before running the project make sure you have installed the proper libraries stated in the README.md file found on the Github page.

In order to run the code in Visual Studio Code you can run the script by typing: *python src/run.py -m predict -v output.csv image1_filename.jpg image2_filename.jpg image3_filename.jpg …* for example, into the terminal and pressing Enter.