## **Data Science Intern Assignment: Adaptive Image Preprocessing for IOPA X-rays**

Company: Dobbe AI

Role: Data Science Intern

Duration: 3-6 months (Possibility of PPO depending upon performance)

### **1. Introduction & Problem Statement**

At Dobbe AI, we are developing cutting-edge AI solutions to assist dental professionals in diagnosis and treatment planning using dental radiographs. Our primary input data consists of Intraoral Periapical (IOPA) X-ray images, typically provided in DICOM or RVG formats.

A significant challenge we face is the variability in image quality across different imaging software and devices used by dental clinics. These variations manifest as differences in sharpness, brightness, contrast, and noise levels. Our current static image preprocessing pipeline struggles to handle this diversity effectively, leading to suboptimal performance of our downstream AI models.

**The Goal:** Develop an adaptive image preprocessing pipeline that can intelligently adjust image parameters (sharpness, brightness, contrast, noise reduction) based on the characteristics of the input IOPA X-ray. This pipeline should aim to standardize the image quality, making them more suitable for subsequent AI analysis, regardless of their original acquisition parameters.

### **2. Objectives of the Assignment**

This assignment aims to assess your ability to:

* Understand and analyze image quality variations in medical imaging.
* Apply image processing techniques for enhancement and noise reduction.
* Explore and potentially implement adaptive algorithms or machine learning approaches for image preprocessing.
* Work with medical image formats (DICOM).
* Structure and document your code and findings.

### **3. Dataset (Simulated/Provided)**

For the purpose of this assignment, you will be provided with a small dataset of IOPA X-ray images. This dataset will deliberately contain images with varying characteristics (e.g., some bright, some dark, some noisy, some blurry).

* **Format:** The images will be provided primarily in .dcm or .rvg (DICOM) format. **Variability:** The dataset will include images simulating different acquisition conditions:
  + Images with varying brightness and contrast.
  + Images with different levels of sharpness/blurriness.
  + Images with visible noise (e.g., salt-and-pepper, Gaussian).
* **Metadata:** DICOM files contain rich metadata. Explore how this metadata might be useful (e.g., Modality, SOPClassUID, PixelSpacing, PhotometricInterpretation).

### **4. Technical Requirements & Deliverables**

#### **4.1. Setup & Environment**

* You are free to use Python as your primary programming language.
* You are free to use any library which helps you to achieve your target.
* Key libraries you might find useful: pydicom, OpenCV (cv2), Pillow (PIL), NumPy, SciPy, scikit-image, scikit-learn, TensorFlow/PyTorch (if pursuing deep learning).
* Ensure your code is well-commented and organized.

#### **4.2. Core Tasks**

1. **DICOM File Handling:**
   * Develop a robust method to read and parse DICOM files, extracting pixel data and relevant metadata.
   * Demonstrate the ability to visualize the raw DICOM images.
2. **Initial Image Analysis & Characterization:**
   * Propose and implement methods to quantify image quality attributes (e.g., brightness, contrast, sharpness, noise). This could involve:
     + **Brightness:** Mean pixel intensity, histogram analysis.
     + **Contrast:** Standard deviation of pixel intensities, RMS contrast, Michelson contrast.
     + **Sharpness:** Laplacian variance, Tenengrad, image gradient magnitudes.
     + **Noise:** Standard deviation in flat regions, wavelet-based noise estimation, or statistical measures.
   * Visualize the distribution of these metrics across the provided dataset.
3. **Static Preprocessing (Baseline):**
   * Implement a simple, static preprocessing pipeline (e.g., fixed histogram equalization, a standard sharpening filter, a basic denoiser).
   * Demonstrate why this static approach fails or performs suboptimally on the diverse dataset.
4. **Adaptive Preprocessing Pipeline Development:**
   * **Algorithm-based Approach** 
     + Design and implement an adaptive pipeline that uses the image quality metrics calculated in Task 2 to dynamically adjust preprocessing steps.
     + Examples:
       - Apply different levels of contrast enhancement (e.g., CLAHE with adaptive parameters) based on initial contrast.
       - Adjust sharpening filter strength (e.g., unsharp masking) based on initial sharpness.
       - Apply different noise reduction techniques (e.g., bilateral filter, NLM denoising) and their parameters based on estimated noise levels.
     + Clearly articulate the logic and heuristics behind your adaptive adjustments.
   * **Machine Learning / Deep Learning Approach** 
     + Propose how a machine learning model (e.g., a regression model, a neural network) could be trained to predict optimal preprocessing parameters for a given input image, or even directly perform image-to-image translation for enhancement.
     + Implement a proof-of-concept. This could involve:
       - Training a model to classify image quality and then apply pre-defined best practices for each class.
       - A simple Autoencoder or U-Net architecture for denoising/enhancement if you can generate noisy/degraded pairs.
     + Discuss the challenges and data requirements for such an approach.
5. **Evaluation & Comparison:**
   * Define quantitative metrics to evaluate the effectiveness of your adaptive pipeline. How will you objectively measure improved image quality (e.g., PSNR, SSIM against a "gold standard" if available, or metrics that indicate suitability for downstream tasks like edge detection)?
   * Visually compare the original images, images processed by the static pipeline, and images processed by your adaptive pipeline.
   * Discuss the advantages and limitations of your adaptive approach.

#### **4.3. Deliverables**

1. **Code Repository:** A well-organized GitHub repository containing:
   * All Python scripts (.py files).
   * A requirements.txt file listing all dependencies.
   * A README.md file (see below).
   * Jupyter Notebooks (.ipynb) are acceptable for initial exploration and visualization but a clear, runnable script for the final pipeline is preferred.
2. **README.md File:** This file should serve as your assignment report and include:
   * **Problem Understanding:** Your interpretation of the problem and its significance.
   * **Dataset Description:** How you handled the provided (or simulated) dataset.
   * **Methodology:**
     + Detailed explanation of the image quality metrics used and their implementation.
     + Description of your static preprocessing baseline.
     + In-depth explanation of your adaptive preprocessing pipeline (algorithms, heuristics, parameters).
     + If applicable, a clear outline of your ML/DL approach, model architecture, and training strategy.
   * **Results & Evaluation:**
     + Quantitative results of your evaluation metrics.
     + Representative visual comparisons (Original vs. Static vs. Adaptive) for various image types.
     + Analysis of your results, highlighting strengths and weaknesses.
   * **Discussion & Future Work:**
     + Challenges encountered and how you addressed them.
     + Potential improvements and next steps for the adaptive pipeline.
     + How this preprocessing step would benefit downstream AI models (e.g., for caries detection, bone loss assessment).
   * **Instructions:** Clear instructions on how to run your code and reproduce your results.

### **5. Bonus Points / Extra Credit**

* **Robustness:** How well does your pipeline handle extreme cases (e.g., extremely dark/bright images, very high noise)?
* **Efficiency:** Consider the computational efficiency of your proposed pipeline. Can it run in near real-time?
* **User Interface (Conceptual):** How might a user interact with or fine-tune this adaptive pipeline? (No need to implement, just conceptualize).
* **Clinical Relevance:** Discuss how misaligned preprocessing could specifically impact dental AI tasks (e.g., false positives/negatives in pathology detection).

### **6. Submission Guidelines**

* Submit the link to your GitHub repository within 2 days of getting the assignment.
* Ensure your README.md is comprehensive and serves as your primary report.
* Be prepared to discuss your solution, thought process, and technical decisions during an interview.

### **7. Evaluation Criteria**

Your assignment will be evaluated based on:

* **Technical Proficiency:** Correctness and efficiency of code, understanding of image processing concepts.
* **Problem-Solving:** Ability to analyze the problem, propose creative solutions, and handle real-world data variability.
* **Analytical Skills:** Depth of analysis in characterizing image quality and evaluating the pipeline.
* **Communication:** Clarity and completeness of the README.md report, ability to articulate ideas.
* **Innovation:** Novelty or sophistication of the adaptive approach (especially for ML/DL integration).
* **Attention to Detail:** Well-commented code, proper error handling (where applicable), and adherence to instructions.

Good luck! We look forward to seeing your innovative solutions.