Update 2 Report: Arsenic Skin Detection Using Machine Learning

CSE 499A: Project Work | Section: 15 | Group: 07 Amit Chandra Das (2014242042), Tanushree Das (2212225042), Ashraful Islam (2212669042)

Introduction & Objective

This project aims to develop a machine learning model capable of classifying skin images as either affected by arsenic poisoning ('infected') or unaffected ('not_infected'). This report details the progress made since Update 1, focusing on data acquisition, initial exploration, and preparation for model training.

Progress Since Update 1

Our project is building a computer program (using Machine Learning) to look at pictures and tell if a person's skin is affected by arsenic poisoning. This report covers what we've done since the last update.

- **Data Acquisition:** The core dataset, "ArsenicSkinImageBD," containing over 8,800 images, was successfully downloaded from Google Drive using the gdown utility within our Google Colab environment. The downloaded archive (ArsenicSkinImageBD.zip) was then extracted.
- **Data Exploration & Verification:**
    - Essential Python libraries (os, numpy, PIL, matplotlib, cv2, sklearn) were imported for data handling and image processing.
    - File paths were verified by listing the contents of the 'infected' and 'not_infected' subdirectories using os.listdir. Sample filenames from the beginning and end of each list were printed to confirm successful loading.
    - The number of images in each class was counted using len(). The dataset was confirmed to be **perfectly balanced**, containing 4446 images for the 'infected' class and 4446 images for the 'not_infected' class. This balance is ideal for model training.
- **Label Preparation (Encoding):**
    - Numerical labels were created for the two classes: 1 for 'infected' and 0 for 'not_infected'.
    - Two separate lists (infected_labels, not_infected_labels) were generated, each containing

the appropriate label repeated 4446 times.

- Label list lengths and initial contents were verified using print().
- The two lists were concatenated into a single master labels list, ensuring the order (infected first, then not_infected) matched the anticipated order for image data processing. Total length and start/end contents were verified.

- **Visual Data Check:** Sample images from both the 'infected' and 'not_infected' classes were displayed using matplotlib to visually confirm the data integrity and appearance.

3. Current Status

The initial phase of data acquisition, verification, and label preparation is complete. The dataset is ready for image preprocessing and subsequent model training stages. The code for these preparatory steps is documented in our project notebook.

4. Next Steps

- The immediate next steps involve executing the remaining parts of our project notebook:
- **Image Preprocessing:** Run the code to resize all images to a consistent 128x128 resolution, convert them to RGB format, and transform them into NumPy arrays.
- **Data Splitting & Scaling:** Split the data into training (80%) and testing (20%) sets using train_test_split and scale the image pixel values to the [0, 1] range.
- **Baseline CNN:** Build, compile, train, and evaluate the initial Convolutional Neural Network (CNN) model defined in the notebook. Visualize its performance.
- **Advanced Model:** Proceed with the implementation and training of the ResNet50 model (potentially with SimCLR pre-training), including necessary debugging.
- **Comparison:** Compare the performance of both models against each other and the benchmark established in the literature review.