Requirements and Installation

**\*Basic knowledge of command line is required.**

Software:

* [Docker](https://www.docker.com/)

Host System:

* 15GB storage
* 8GB RAM
* Modern CPU

Installation Instructions:

1. Download and install the Docker command line environment:  
     
    sudo apt-get install docker-ce docker-ce-cli containerd.io  
   1. Alternatively, you download [Docker Desktop](https://www.docker.com/get-started) if you prefer a desktop GUI.
2. Download [ark\_mirai](https://hub.docker.com/repository/docker/harrivle/ark_mirai/tags?page=1&ordering=last_updated):  
     
    docker pull harrivle/ark\_mirai  
   1. If needed, you can add a suffix with a colon to download a specific version, i.e. “harrivle/ark\_mirai:0.5.1”
3. Verify that the image has been successfully loaded:  
     
    docker image ls  
     
   You should see “harrivle/ark\_mirai” under the “REPOSITORY” column.

Local Command Line Demo

The “ark/mirai” container creates a server for an API. The API implements the endpoint “/dicom/files” to receive DICOM files through a HTTP POST request. The API will then return predictions with the server’s response.

Running the Server/Model:

1. Download and extract [demo data](https://drive.google.com/file/d/1aiwXn0cHIPzVS1eYwJDERijnYg94sIhH/view?usp=sharing).
2. Choose a [PORT] to expose, i.e., 5000. Run the docker image:  
     
    docker run -p [PORT]:5000 harrivle/ark\_mirai  
     
   You should see logging messages indicated that a server has been started.
3. With the extracted folder “mirai\_demo\_data/” in the same working directory, you can send a request to the server to run the model using the “curl” command like so:  
     
    curl -X POST -F 'data={}' -F 'dicom=@mirai\_demo\_data/mlor2.dcm' -F 'dicom=@mirai\_demo\_data/mlol2.dcm' -F 'dicom=@mirai\_demo\_data/ccr1.dcm' -F 'dicom=@mirai\_demo\_data/ccl1.dcm' http://localhost:[PORT]/dicom/files  
     
   The curl command may need to be installed with your package manager depending on your OS. Be sure to replace [PORT] with your chosen port number.  
   1. **Important Note**: Each request to this endpoint should be the equivalent of a single exam. The model requires **four files** from the **same** exam corresponding to different mammography views.
4. After a half a minute or so, you will receive a JSON response like so:  
     
     
   1. On the hospital side, some program/script would need to be written to send files to the API and handle the JSON return object. This program ideally will be what interfaces with a database to grab the DICOM images and store the JSON predictions.

API Documentation

POST /dicom/files – Takes four .dcm files belonging to a single breast exam as input and returns probability diagnosis across five years.

Request JSON:

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| data | JSON | Required freeform JSON object. Any data contained in the object will be returned in the HTTP response JSON. Can be empty. |

Response JSON:

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Subfield** | **Type** | **Description** |
| data |  | JSON | Data containing the `predictions` object i.e. {“predictions”: {…}} |
| metadata |  | JSON | Freeform JSON object taken from the `data` field in the request JSON. |
| message |  | string | A message detailing any important information (usually regarding error messages). |
| statusCode |  | int | The HTTP status code of the response. |
| runtime |  | string | Runtime string in seconds, formatted as “0.00s”. |