TECH4HERITAGE

Road Map of the solution

By UnderDogs

Members:
Cherry Jain
Rishab Shekhar
Sarthak Chauhan
Shriram Rawat
Vikramjit Dutta

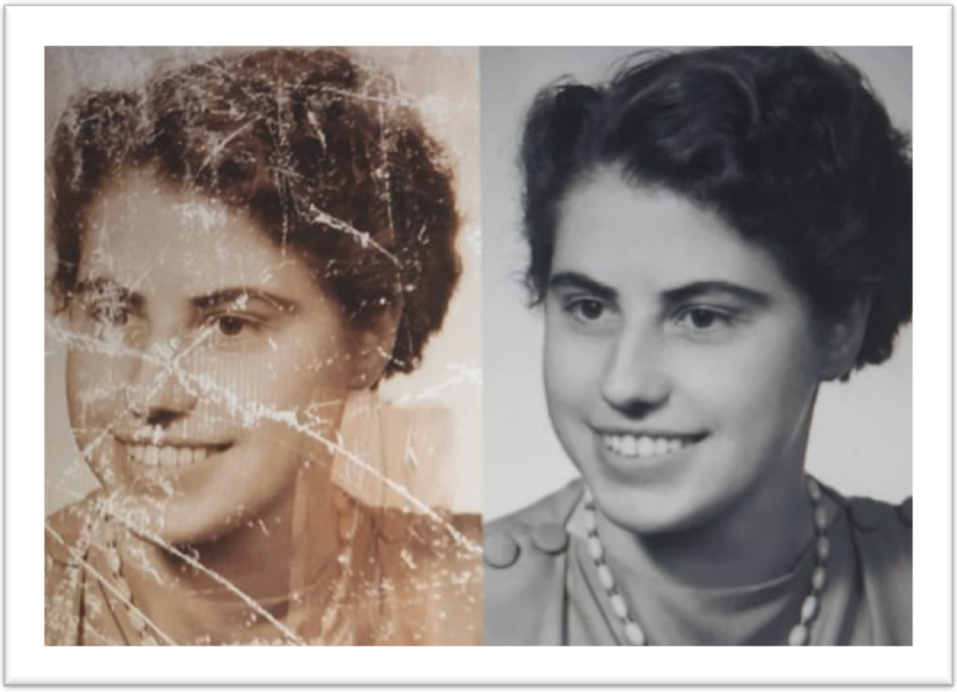
Language used: Python

Functions/Technologies: Keras, Tensorflow, Pytorch, Opencv (other functions may get added up)

Jupyter, Google Collab: Not using

Platform being used: Visual Studio/Spyder

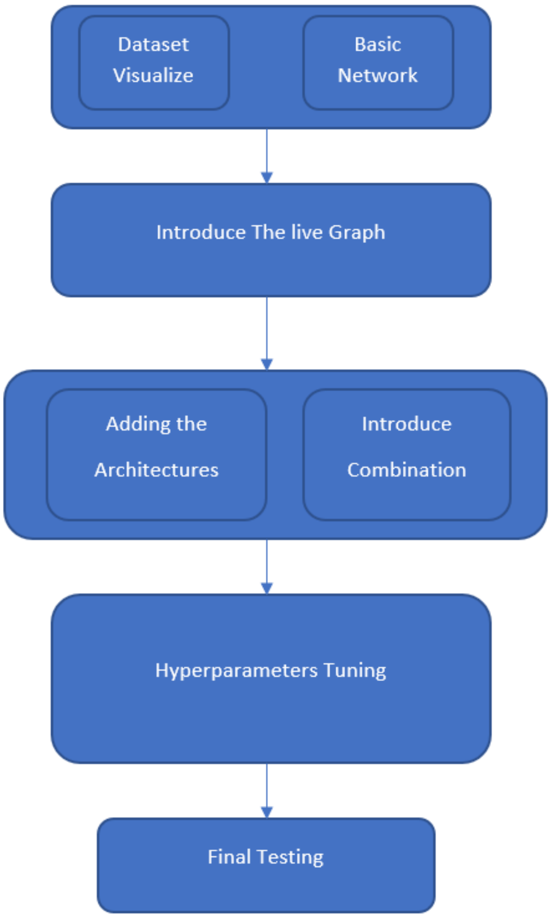
INPUT: OUTPUT Transformation ->



How do you plan to achieve completion of the project?

The project cab be broken into 2 parts:

1) Dataset generation
2) Serialization of Neural Networks
3) Network Tuning and visualization



Dataset generation

Since Beginning with a small dataset,

- we need to generate different types of images for training dataset using Opencv, for example, blurred, high hue, contracts, ….
- Increasing dataset size with ImageDataGenerator, which will help in rescaling, creating rotated, flipped, zoomed images.

Serialization of Neural Networks

- Taking a common approach starting with a fully Connected Neural Network to see as to where the current network stands.
- Adding new network layers

Testing

- Dataset testing on various models, to clear out what is required to be extracted and what parameters to be excluded

Hierarchical Approach

(The idea is to discuss and build the layout during the Project period. Even scraping the current proposed designs if need be)

The proposed time line can be segmented as:

Initial day:

- Plan how to implement Project serialization for external nested timelines.
- Build the basic Neural Network with a Long Short Term Memory layer, to get the idea where the basic Network stands before moving into deeper approach.
- Separate function to Generate images, by using concepts mentioned above.

- Plan the different approaches if necessary.

Next Day:

- Introduce Live graph
- So, the basic graph displays the Losses and Accuracies of the Functions after the work is completely done, but what if we require to keep the track of these values during the process is going on so the Live graphs comes to play in this.
- This is also help to see if the any major changes are required, we can easily add in the early stopping function if the accuracy starts to go off due to some reasons.

Following Days:

- Adding in the various well-known Neural Network architectures to move a little bit further in the deeper concepts.
- Architectures may include Resnet, Unet, Squeeze Net, Alex Net, ……
- Initially, they will be serialized and nested to see their performances for our objectives.
- These architectures used will be added with pre-defined weights.

Remaining Period:

- Beginning with the Tunning of the parameters for our Networks.
- So basic version of hyperparameter is using Hyperband or keras.tuner. But we are going to create our own Neural Network Tuner, with basic features like the following:
    - Adding different number of layers
    - Adding number of neurons/units.
    - Adding Different number of loss functions, different layers, activation functions, ……
    - Adding Different number of Dropout values and layers.
    - Introducing Early stopping is required
    - Adding multiple learning rates
- For Better results or for improvement combining the architecture is the best method, like:
    - U-Net + U-Net;
    - U-Net + Squeeze Net + Alex Net, ….
- Combining Architecture is seen to increase result accuracy.

Final Testing

Last thing that only remains is the Testing period and solving the errors occurring throughout the code.