Automated method for taking Electric Meter Reading Based on Image Processing and custom CNN classifier

Abstract

With the increase in demand of electricity, it is very crucial to keep accurate track of electricity usage and store data collected from electric meters in digital format for later analysis and billing process. In many countries process of taking the reading of electric meters is still manually performed by human employees which not only involves the risk of human error but also inefficient in terms of time. The aim of this project is to develop an application which will automate the process of meter reading through image processing and store meter's data automatically into a database. User will be provided with GUI through which he can input a video clip of electric meter. After applying a bunch of image preprocessing techniques image's portions with meter reading and meter number are extracted. These extracted portions are then passed to trained model and google tesseract respectively for digits recognition and final results obtained are displayed to user and stored in the database. The model used for classification of LCD's digits is based on Convolution neural network with 2 sets of Conv and max pool layer followed by one dense and one output layer. This model gave us an accuracy of 96 percent, 92 percent, and 90 percent on training, validation, and testing data, respectively.

Contents

Contents

1. Introduction

Electricity has become one of the most fundamental need of humans in the current era. Every electronic device that we use on daily basis such as mobile phones, television, refrigerators, electric cars, etc. requires electricity. With the increase in demand for electricity, it is very crucial to keep track of electricity usage and charge customer accordingly. In order to track down consumer’s electricity usage, electric meters are used. Although developed countries are using smart electric meters which automatically report back used electricity units but still many countries are using conventional non-smart electric meters where the process of taking meter’s reading is still manually done by humans. This human involvement increases the risk of error in the meter reading process. Furthermore, in the current age of computers, it is also very important to store data in digital format which will not only help in computerized billing but also use in other fields like data science to extract different kind of useful information.

Generally, they are two main patches of information which should be collected during the process of meter reading. First, one is Currently used electricity units and the second one is meter Number that identifies the meter’s owner. Currently used units are displayed on a seven segmented LCD which is situated in the middle of electric meter. Below this LCD some descriptive information is written in black font on white background. Among this information meter Number is written with prefix “No.” normally these readings are taken by human employees and later on entered into database manually. The aim of this project is to automate the process of reading meters and develop an application that will take a clip of video as an input and automatically read Currently used electricity units and meter no and store them in the database.

Although OCR commercially in market like google tesseract are very effective in recognizing digits but when It comes to recognize digits in seven segments display, they are not very accurate. Digits of this type are composed of seven segments with a bit of gap in between them moreover structure of these digits is quite different than the one we write or type. Hence LCD digits are not well recognized with these traditional OCRs. Therefore, we decided to develop convolution neural network for recognition of these seven segmented display digits. CNN is proved to be very effective in the recognition of letters and digits. They are best known for their feature extraction ability and based on these extracted features CNN can classify given image into the classes of digits. CNN is not only effective but also computationally efficient as compare to other machine learning algorithms.

In general, we have trained a CNN model on a given dataset and deploy it in an application when user uploads video of meter bunch of preprocessing techniques are applied and meters number and meter reading are extracted. Meter reading is recognized by trained model and meters number is recognized by google tesseract. Once meter number and meter reading are recognized results are shown to the user and store in the database. The detail of these processes is defined in the methodology section.

2. Dataset

Dataset use in this project is self-created. Originally there were 120 images of electric meters from which images for different classes were extracted. Initially, dataset contained 600 images where there were 11 classes each having 40-60 images. Ten classes are of number from 0-9 and 11th class represent blank space. Since dataset was very limited hence data augmentation was applied to increase data volume. After applying data augmentation dataset volume became 1433. Beside data augmentation other preprocessing techniques were applied to make image text more prominent from there background. Image count for different classes is shown below.

Classes	image count
zero	174
One	168
Two	121
three	147
four	126
Five	102
six	123
seven	111
eight	129
nine	126
blank	106

3. Methodology

The main backbone of proposed system is convolutional neural network (CNN). Deep learning specifically Convolutional Neural Network are proved to be every effective in image's features-based classification. CNN are not only effective but also computational efficient as compare to other machine learning algorithm such as artificial neural network. Hence CNN is used in this project for character recognition of seven segmented display of electric meters.

Our Proposed system has two main stages, in first stage bunch of preprocessing techniques were applied to make image clearer and more suitable for classification. Main objective of this stage was to extract and segment LCD portion of electric meter and meter number portion of meter. Second stage involves designing and implementing of convolutional neural network that trains on collected dataset and learns to classify given image into classes of digits. Detail of these stages and techniques used in them are define below. Figure1 show an general overview of whole system.

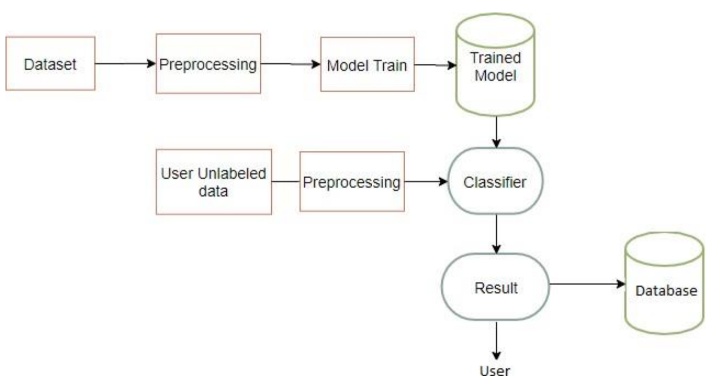


Figure 1 General overview of system

3.1. Preprocessing stage

Preprocessing stage involves all method that are applied on original frames of video feed coming from source to extract LCD portion and meter no portion of image and to enhance the features of these extracted portion and make them clearer for classification phase. To make system efficient in terms of computation time One frame from every two frames was selected for processing. In the beginning these frames were aligned according to a standard template to make sure area of interest in image are at right place for extraction. After this alignment of frame was checked to exclude those frames which are not properly aligned to save some computation time. Once we have image in which meter is in correct orientation same as that of standard template, we extract LCD portion and meter number written in information section of meter. After extraction of areas of interest from image, preprocessing techniques such imaging thresholding, image smoothing etc. were applied followed by morphological transformation to make text more prominent and clearer. Lcd portion is segmented into small portion so that each portion contain single character. This segmented LCD image is passed to trained model for classification whereas meter number portion is passed to google tesseract for number recognition.

Figure 2 Original frame

3.1.1. Image alignment

Feature based image alignment is used in this project for which set of features are detected in sample image. These features are than matched with features of new image based on which transformation is calculated that wraps new image to sample image. At first Oriented Fast and Rotated BRIEF (ORB) feature detector is used to detect features in sample image and new image. Once features are localized their descriptors are computed. After that, these features from new and sample images are matched together by calculating hamming distance between them which is shown in figure 3. Based on these calculated distances Matches were sorted and out of 600 matches only 0.1 percent good matches were kept. Once good matches are found Random Sample Consensus (RANSAC) based homograph is calculated. This Calculated homograph was applied on new image to map it to sample image. Resultant image after image alignment is showing in figure in figure 4. [1]

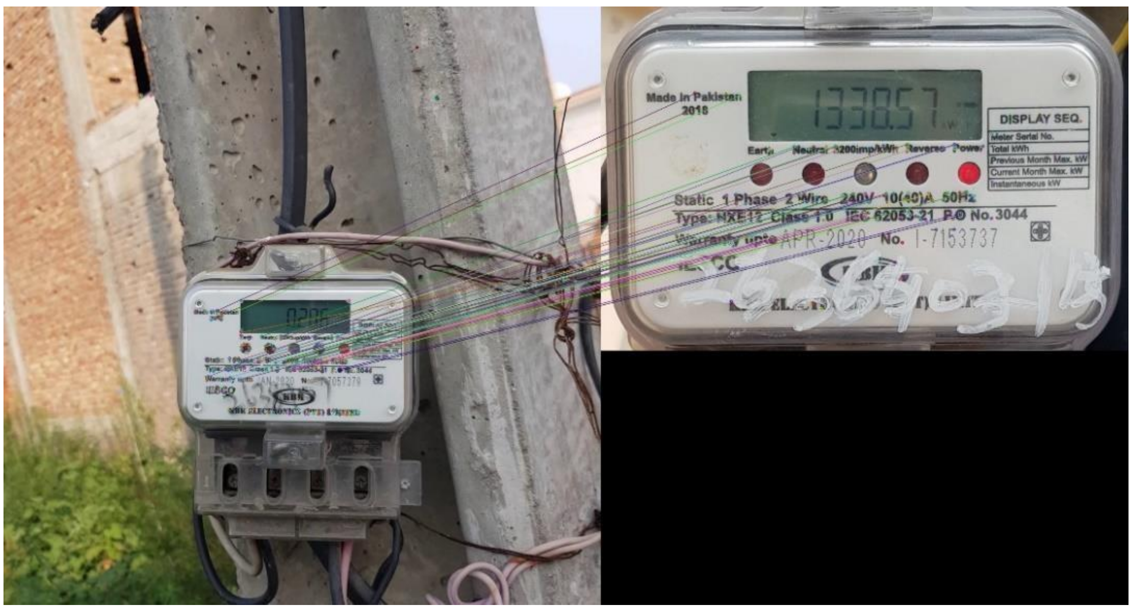


Figure 3 feature matching between original frame and sample image.

Figure 4 After image alignment

3.1.3. Extracting area of interest

There are two main area of interest in meter image, first one is meter's LCD which show current reading of used electricity Second one is meter number portion that is basically the identity of meter. In order to extract lcd, at first image's edges were detected for which image was converted into grayscale than noise was removed from image by applying gaussian blur after which canny edge detection algorithm was used to detect edges in image. Once the edge become prominent, we find largest rectangle in image which is basically meter's LCD for this, At first contours were detected in image. these contours were sorted according to area so that contours with largest comes in front of list. This list was iterated to find contour with 4 vertices. After which four-point perspective transform is used to extract this contour. Since after image alignment and resizing image is almost identical to standard template in terms of size and orientation hence LCD extracted from every frame is same in terms of its size therefore a loop was used to segment extracted LCD according to fixed coordinates. Similarly, meter number portion is extracted by using hard code coordinates. [2]

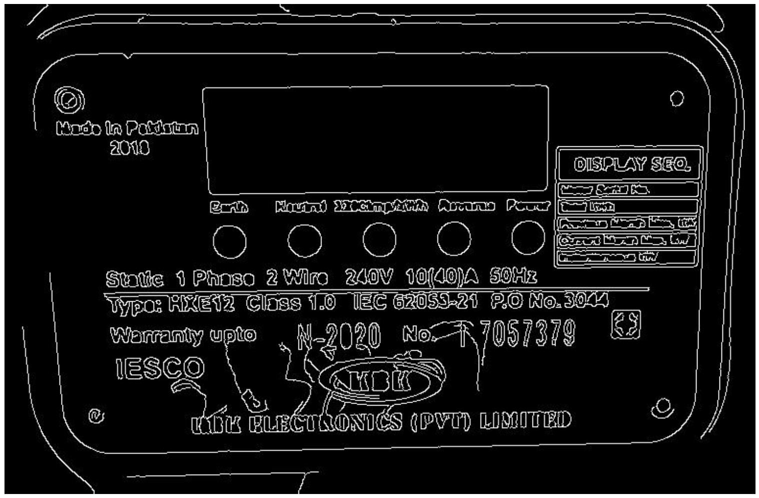


Figure 5 Edge detection

Figure 6 Extracted LCD.

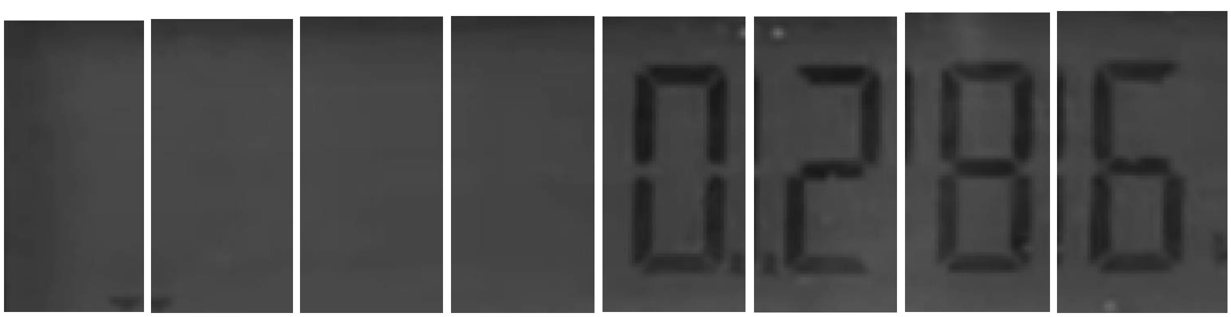


Figure 7 Segmentation of extracted LCD

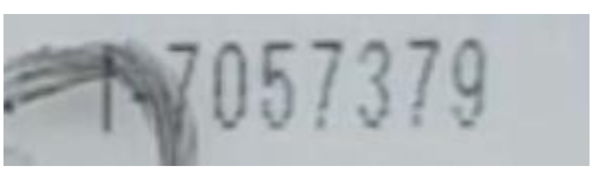


Figure 8 Extracted meter number.

3.1.4. Preprocessing and Morphological transformations

After area of interest are extracted next stage is to apply different preprocessing techniques to make numbers more prominent and clearer. While preprocessing segmentate LCD portion at first Gaussian Blur was applied to remove noise from images than thresholding was applied to make digit distinguished from background. Since images are bimodal and have two peaks in histogram hence Otsu's Binarization was used which compute thresholding value automatically. After thresholding dilation was applied to fill gaps between body of individual digit. At the end these segmented preprocessed lcd portions were passed to trained model for classification. If we talk about meter number portion adaptive thresholding was applied at first to make text more prominent than background than opening was applied to remove noise left after thresholding. At the end dilation was applied to give text a consistent look. Once the dilation was applied meter number portion was passed to tesseract for number recognition.

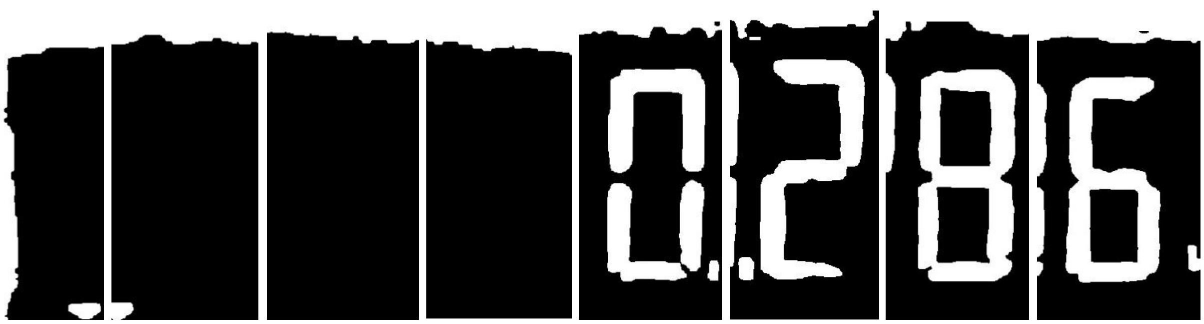


Figure 9 segmented LCD after preprocessing

Figure 10 meter number portion after preprocessing

3.2. Model training

The dataset used in this project is very small and contain only 1433 images in total furthermore dataset contains noisy images hence in order to avoid overfitting we start with simplest convolutional network with 2 convolutional layers only as used by Yann LeCun et al [3] in LeNet-1. At first dataset gray scale images were resized to 50 x 50 so that each input image had a dimension of 50x50x1. Than these images were transform into matrix format after that image data was normalized at the end data was split into 90 percent training and 10 percent testing set. Out of this 90 percent training data again 10 percent data was carved out for validation set.

If we talk about network architecture at first, we used only two set of convolution layer and max pooling layer as used in LeNet-1[3]. First Set contained a convolutional layer with 32 filters having kernel of dimension 3x3. Relu activation function was used at first conv layer. Max pooling layer used in first set had pool size of 2x2 and stride 1. Second Set contained a convolutional layer of 32 filters with kernel of size 3x3 . Relu was used as activation function in second conv layer. Max pooling layers used in second set also had pool size of 2x2 and stride 1. At end a dense layer of 128 units was added form which 0.5 was dropped and relu activation was used followed by dense layer of 11 units with softmax activation function. We trained this architecture for up to 1000 after which training accuracy approached to 1 but validation set accuracy remains at 90 percent which indicated that model was overfitting therefore, we again trained this model but this time we trained it for 600 epoch which gave us an accuracy of 96 percent on training set and 92 percent accuracy on validation set. After trying 2 convolutional layer network we increase complexity of model by adding another convolutional layer of 64 filters and trained this model for 700 epochs. This new architecture gave us 0.985 accuracy on training set and 90 percent accuracy on validation set hence this huge gap showed that increasing complexity was of no use since it was causing overfitting. Considering all these finding was decided to use 2 convolutional layer model which gave us highest accuracy of 92 percent on validation set. At end upon testing this 2-conv layer model on testing dataset it gave us an accuracy of 90 percent.

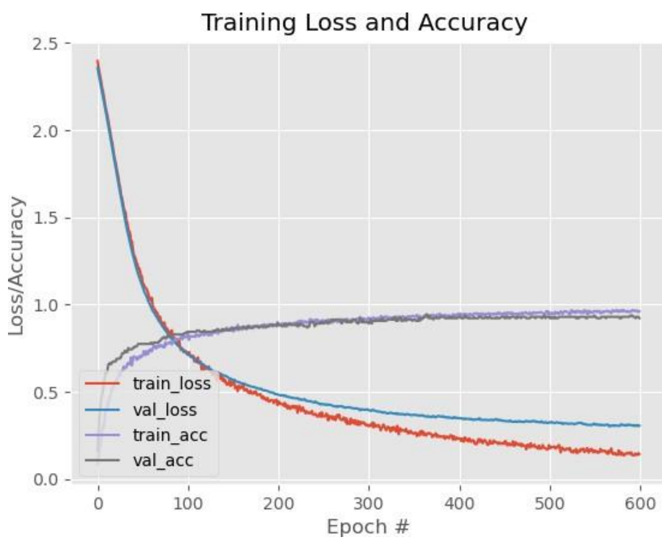


Figure 11 training and validation accuracy graph

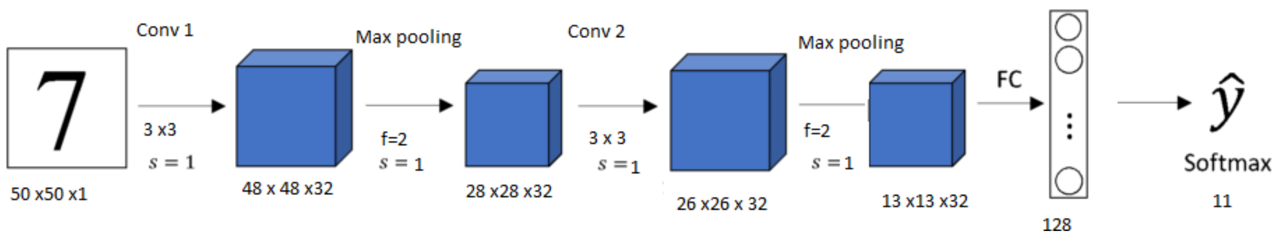
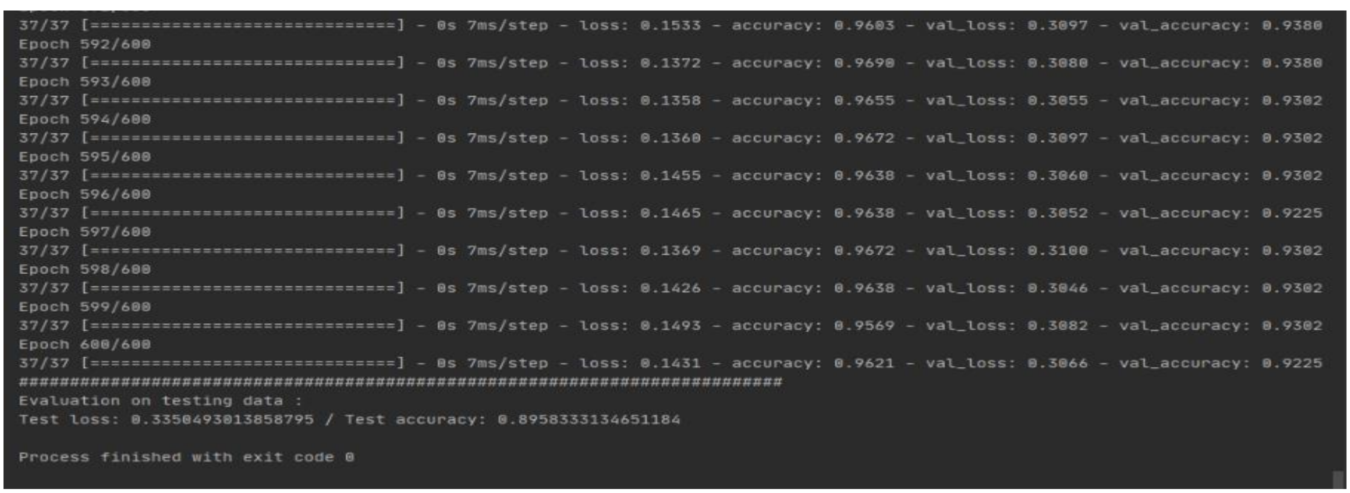


Figure 12 Network architecture



```
37/37 [==============================] - 0s 7ms/step - loss: 0.1533 - accuracy: 0.9603 - val_loss: 0.3097 - val_accuracy: 0.9380
Epoch 592/600
37/37 [==============================] - 0s 7ms/step - loss: 0.1372 - accuracy: 0.9690 - val_loss: 0.3080 - val_accuracy: 0.9380
Epoch 593/600
37/37 [==============================] - 0s 7ms/step - loss: 0.1358 - accuracy: 0.9655 - val_loss: 0.3055 - val_accuracy: 0.9302
Epoch 594/600
37/37 [==============================] - 0s 7ms/step - loss: 0.1360 - accuracy: 0.9672 - val_loss: 0.3097 - val_accuracy: 0.9302
Epoch 595/600
37/37 [==============================] - 0s 7ms/step - loss: 0.1455 - accuracy: 0.9638 - val_loss: 0.3060 - val_accuracy: 0.9302
Epoch 596/600
37/37 [==============================] - 0s 7ms/step - loss: 0.1465 - accuracy: 0.9638 - val_loss: 0.3052 - val_accuracy: 0.9225
Epoch 597/600
37/37 [==============================] - 0s 7ms/step - loss: 0.1369 - accuracy: 0.9672 - val_loss: 0.3100 - val_accuracy: 0.9302
Epoch 598/600
37/37 [==============================] - 0s 7ms/step - loss: 0.1426 - accuracy: 0.9638 - val_loss: 0.3046 - val_accuracy: 0.9302
Epoch 599/600
37/37 [==============================] - 0s 7ms/step - loss: 0.1493 - accuracy: 0.9569 - val_loss: 0.3082 - val_accuracy: 0.9302
Epoch 600/600
37/37 [==============================] - 0s 7ms/step - loss: 0.1431 - accuracy: 0.9621 - val_loss: 0.3066 - val_accuracy: 0.9225
###########################################################################
Evaluation on testing data :
Test loss: 0.3350493013858795 / Test accuracy: 0.8958333134651184

Process finished with exit code 0
```

Figure 13 Training, Testing and validation accuracies

4. System Testing

Model that we choose to deploy in application showed an accuracy of 90 percent on testing dataset which is consist of 10 percent images of original dataset. This application take video feed and recognize data in one frame out of every two frame until result with frequency of greater than 3 is found. Upon testing this application on completely different distribution it showed remarkable result. We took 11 video feeds as real-world testing scenario out of which application correctly recognize 10 videos. Although model testing accuracy was only 90 percent on testing data but in application since we are getting multiple frames hence overall application was able to perform better in real world use case.

5. Graphical User Interface

A user-friendly graphical user interface was designed to provide user a medium to effectively interact with system. While developing user interface all Human computer interaction principles were followed to make application functional and to ensure positive user experience. Application allow user to upload video clip from device. Visuals of selected video and results are shown to user. User is also provided with functionality to change results before storing to database. Throughout application interface user is kept updated regarding status of system. Overall application was designed to avoid complexity and provide user a simple design which is essay to use.

5.1 Startup window

As user starts up application, he is greeted with window show in figure 14. This is the main window of application and provide all major functionalities of application. At this stage result section show not available (NA) sign and color of start button is yellow indicating not ready yet as no file is selected.

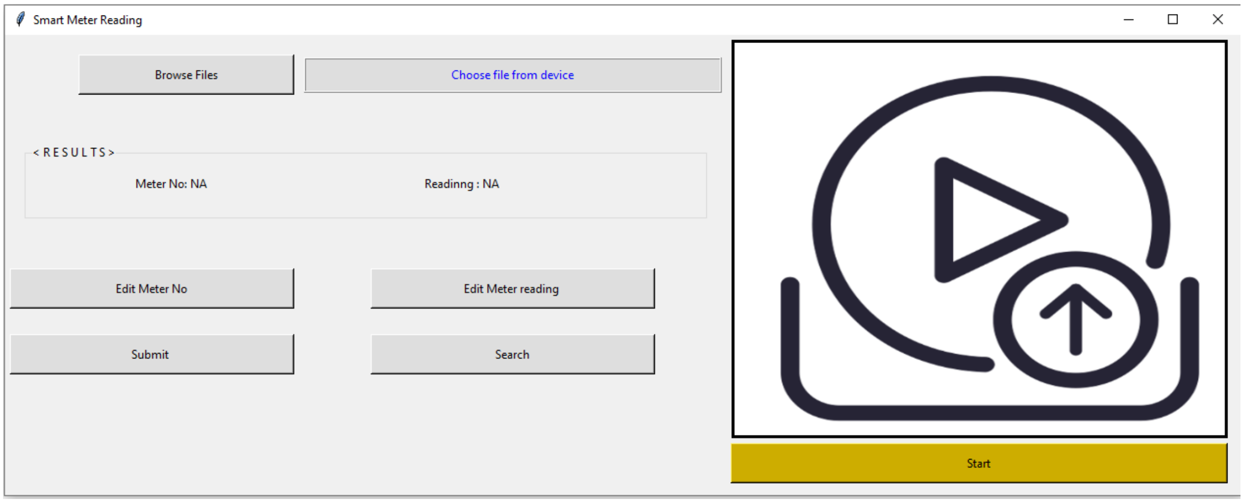


Figure 14 Startup window

5.2 Selecting File

To select file user can press “browse file” button to select any file from device. Once file is selected its path is displayed to user in file explore portion and first frame of video is also displayed to make sure user is choosing file he is intended to. At this point color of start button turn into green showing application is ready to process given video and user can press start button.

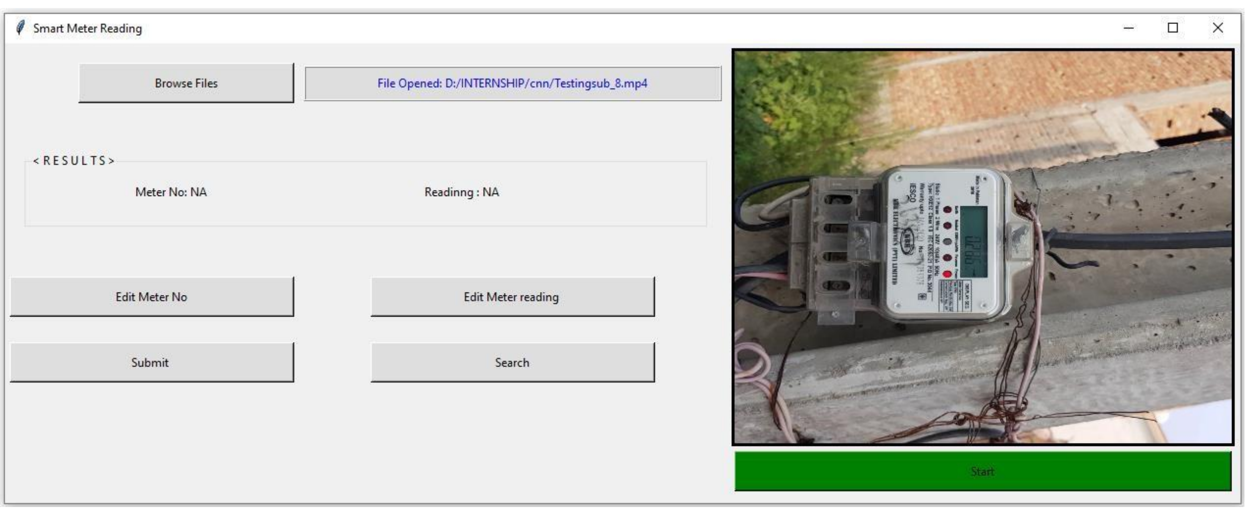


Figure 15 After selecting file

5.3 Waiting state

Once user press start button application goes into processing state. Start button text changes to "processing" and color changes to red indicating that application is in processing state and won't take any user input. Furthermore, and a time processing logo appears on given file image to indicate that user file is being processed.

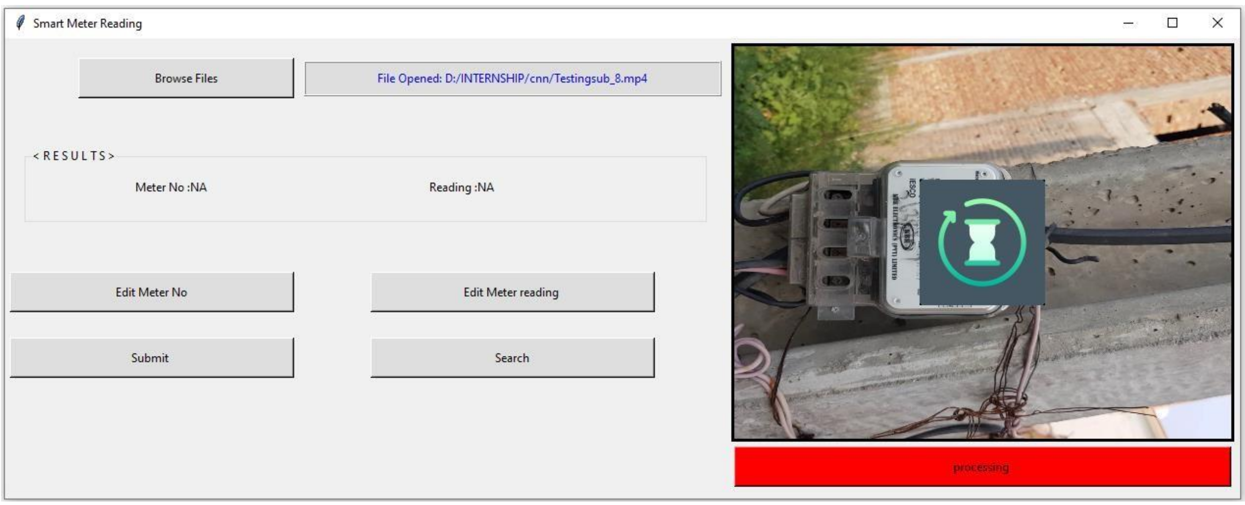


Figure 16 processing state

5.4 Showing output

Once the file is processed a visual image of meter is displayed with detected LCD and recognize digits. Furthermore, recognize reading and meter number is also displayed to user in text form in results portion.

Figure 17 Window showing output.

5.5 Editing result

After application finishes processing and shows result on screen user has option to change either meter number or meter reading or both. Once user click on edit button a pop-up screen will appear which will ask for new value. In this way user can correct values if application fails to recognize digits properly.

Figure 18 Changing recognize values manually.

5.6 Storing result in Database

Once the user cross check recognize value with actual value, he can store result in database by simply clicking on submit button.

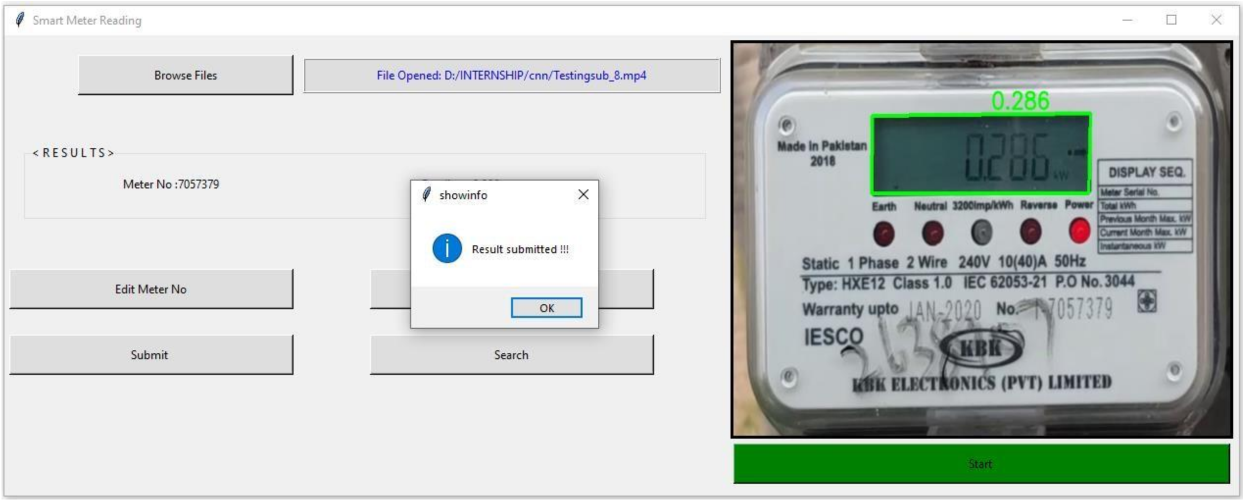


Figure 19 Storing results in database

5.7 Searching database

User is also provided with functionality to search database for previous result. Once user enters any meter number all reading ever taking against that meter number will appear on screen.

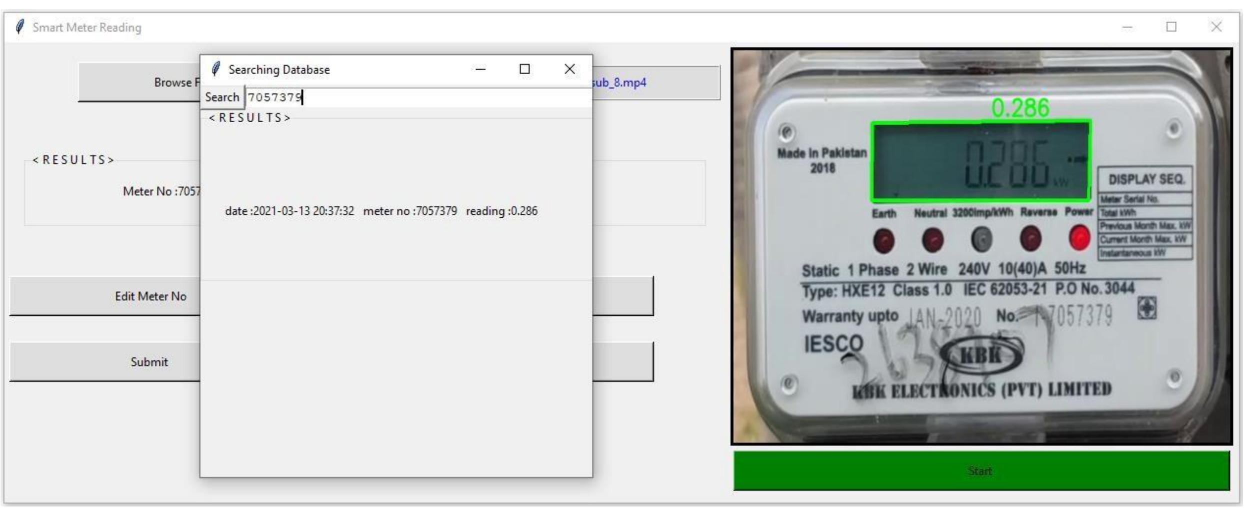


Figure 20 Searching Database.

References

[1] Mallick, S. (2021, March 11). Feature based image alignment using opencv (C++/Python). Retrieved March 13, 2021, from https://learnopencv.com/feature-based-image-alignment-using-opencv-c-python/

[2] Andrei, et al. “Recognizing Digits with OpenCV and Python.” PyImageSearch, 18 Apr. 2020, www.pyimagesearch.com/2017/02/13/recognizing-digits-with-opencv-and-python/.

[3] LeCun, Yann, et al. "Comparison of learning algorithms for handwritten digit recognition." International conference on artificial neural networks. Vol. 60. 1995.