INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2024, VOL. 70, NO. 3, PP. 537–544
Manuscript received October 31, 2023; revised July, 2024. doi: 10.24425/ijet.2024.149576

Optimized Supervised ML for Medicinal Plant Detection - An FPGA Implementation

Amrutha M. Raghukumar, Gayathri Narayanan, and Geethu Remadevi Somanathan

***Abstract*—Medicinal plants have a huge significance today as it is the root resource to treat several ailments and medical disorders that do not find a satisfactory cure using allopathy. The manual and physical identification of such plants requires experience and expertise and it can be a gradual and cumbersome task, in addition to resulting in inaccurate decisions. In an attempt to automate this decision making, a data set of leaves of 10 medicinal plant species were prepared and the Gray-level Co-occurence Matrix (GLCM) features were extracted. From our earlier implementations of the several machine learning algorithms, the k-nearest neighbor (KNN) algorithm was identified as best suited for classification using MATLAB 2019a and has been adopted here. Based on the confusion matrices for various k values, the optimum k was selected and the hardware implementation was implemented for the classifier on FPGA in this work. An accuracy of 88.3% was obtained for the classifier from the confusion chart. A custom intellectual property (IP) for the design is created and its verification is done on the ZedBoard for three classes of plants.**

***Keywords*—Machine learning, GLCM, FPGA, Intellectual Property, Medicinal Plants**

I. Introduction

MANY of the medicinal plants are at the verge of extinction among the thousands of ayurvedic medicinal plants available in the world. These can be used for the treatment of various disorders such as fertility issues, respiratory disorders, cardiac problems, excretory issues, rheumatoid arthritis and joint pains, to name a few. Therefore, the accurate identification and protection of these medicinal plants are of great significance, especially in today's times. The physical or manual identification can be carried out but it would be an excruciatingly slow and difficult process. Manual identification would also require a lot of experience and exposure, which can take years to gain. Also, many such plant species are located in places which are inaccessible. Hence, an intelligent approach to identify medicinal plants is essential. If such a system is mounted on a drone, it can identify the image taken by the camera and classify the plant appropriately. Images of the leaves of selected medicinal plants are considered in this work since leaves are generally available on the plant almost all year round for Indian climatic conditions. Due to the unavoidable development of urban areas, many of the Ayurvedic medicinal plants are at the verge of extinction. These plants are the main source of medicines which can prevent, treat and cure various disorders, which may otherwise not be satisfactorily treated using allopathic approaches. Due to the long-lasting benefits of adopting the Ayurvedic approach, the identification and conservation of medicinal plants have become a dire necessity. In this work, we attempt to develop a system to automate the medicinal plants detection starting from the compilation of the data set. In order to build our data set, the leaf images were captured with a camera from a herbal garden Amritavanam situated at Amrita School of Ayurveda, Amritapuri Campus. The images captured were resized and the features are extracted using MATLAB platform. These features were then provided as inputs to the hardware description language (HDL) software. Further, the classification is done using the k-nearest neighbour (KNN) algorithm. Here the data set is labelled beforehand and hence our approach is considered as a supervised machine learning (ML) classifier. The superiority of the KNN is validated by a performance comparison undertaken by the authors in [12]. The optimized value for k is chosen as three, since, for the chosen value of k, the classifier shows better accuracy value when simulated in MATLAB. This approach aims to render a smooth system of identification and conservation of Ayurvedic medicinal plant species and address the availability and yield of such plants based on the requirement. Following the brief introduction to the work in this section, the following sections are organized as follows. Section 2 lists out the related literature works followed by the System Methodology and relevant parameters in Section 3. In Section 4, an overview of the IP block that helps to customize the hardware implementation is explained. Section 5 details the System Implementation of the work explaining how the feature inputs are integrated into the FPGA block. Section 6 discusses and analyses the results and Section 7 summarises the work and presents the conclusion.

Amrutha M. Raghukumar is a DFT Engineer at Anora Semiconductor Labs Pvt Ltd, Bengaluru, India. (e-mail: amruthamr@anoralabs.com).
Gayathri Narayanan and Geethu Remadevi Somanathan are with the Department of Electronics and Communication Engineering, Amrita Vishwa Vidyapeetham, Amritapuri, India (e-mail: gayathrin, geethurs@am.amrita.edu).

II. Related Works

The authors of [1] compared the machine learning algorithms like Support Vector Machine (SVM) and KNN for the automatic detection of the medicinal plants. The paper considered the images of leaves of ten separate species. Images were captured corresponding to different orientations. The GLCM based textural features, color features and shape features are considered and an accuracy of 100 % and 93.23 % respectively were obtained for KNN and SVM classifiers. Medicine plant classification using texture analysis is implemented in [2]. In [3], the authors have applied a similar approach as [2] and [3] for Romanian medicinal herbs. The authors in [4] implemented a method to correctly predict the plant class. They considered six species of plants for the classification purpose, and used artificial neural network (ANN) as the classifier. Eight input features including the shape, texture and color features were taken. The leaf images were considered and an accuracy of 94.4 % was obtained. An approximate 17.24 % reduction in computation time was also observed. The authors in [5] used high level synthesis (HLS) techniques and an IP core design for KNN hardware acceleration, evaluated on the FPGA board. The authors implemented approaches like pipelining and parallelism and AXI-Master Interface which is memory mapped to implement on FPGA. The advantage of this design approach is that it is about 35 times faster than general purpose processors (GPP)-based implementation and it reduces the cost as well as the development complexity.

In [6], authors proposed an architecture for parallel computation of GLCM. They make use of the HLS tool for implementation and considered four GLCM matrices for four angles - 0, 45, 90 and 135 degrees. The implementation is done entirely on hardware ZedBoard which has a peripheral logic and an ARM Cortex A9 processing system integrated on it. An optimization in order of 33 % in latency number is observed in the validation results when compared to the literature surveys they conducted for their work. In [7], the hardware realization of KNN classifier is done. They make use of pipelined and parallel architecture based on Predetermined Range Search (PRS). They also discuss a technique to compute the reference distance. The implementation was done on Virtex 4 FPGA which was a simple design and they obtained a clock frequency of about 186.4 Mhz, but the BRAM coverage got reduced by one third. Authors of paper [8] used a computer vision algorithm for identification of medicinal plants. In this method, the image pre-processing is done such that conversion from color to gray scale and edge detection is done. Then the feature extraction is done for five different species of neem plant in particular. The authors also discussed about the Probabilistic Neural Network and Support Vector Machine classifiers. In the paper [9], authors considered the paddy leafs for the detection of the diseases affected on them. The authors calculate the percentage of infected plant area to understand how much severely disease is affected and also to determine the type of the disease. So that proper pesticides can be applied and control the spreading disease. In this paper [11] authors considered ten species of plants where some plants belong to the same family. The leaf images are considered for automatic identification of plant species. The image segmentation is done on the captured images. Then several features such as roundness, rectangularity, area of leaf, color features etc are considered for feature extraction process and the classification was done by fixing proper boundary for each class. Authors of [10] consider the flower images for detection of herbal plants.The segmentation is done on the flower images using Otsu’s thresholding. The features considered for feature extraction includes color, texture and shape features classified using SVM, KNN and Decision Tree algorithms. The performances of these classifiers are compared among them based on the accuracy. In this work, we attempt a hardware realization of the supervised ML algorithm applied for classification of medicinal plants using FPGA implementation. Following the introduction and brief review of relevant techniques in this area, in the following sections we present the proposed method, IP block, implementation method and a discussion of the results obtained.

III. System Methodology and Parameters

This work is carried out in three phases namely image acquisition, feature extraction and classification. The following subsections will briefly decribe how the three phases have been addressed in this article.

A. *Image Acquisition*

The leaf images were captured using a 13 mega pixel camera as in [12] maintaining constant distance between the camera and the leaves (both front and back images), considering various orientation and light conditions. Resizing of the captured images were done ensuring no information loss. The plant leaves of medicinal plants such as Carissa, Plectranthus, Clerodandrum, Maloora, Psidium guajava, Ocimum, Hibiscus, Samadera, Azadirachta and Morus are considered in this work as data samples. A total of 650 trained images and 260 test images are available for the work belonging to 10 species of plants. Fig. 1 shows a representative sample set of the leaf samples taken in different orientations, back view, front view and also in different intensities of daylight in white background.

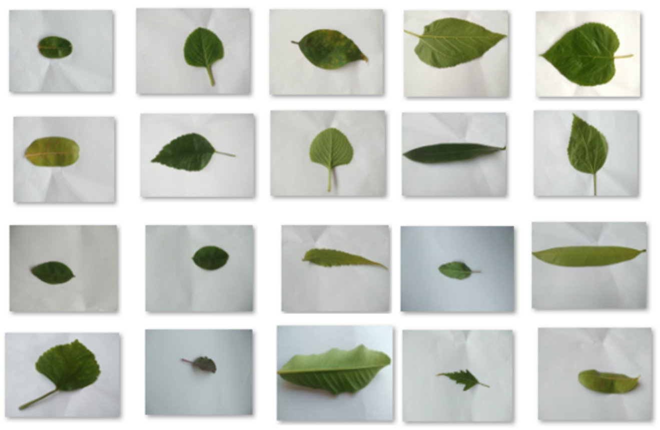


Fig. 1. Image Samples

B. *Feature Extraction*

Feature extraction is an important step and is required when the data size is to be reduced. In case of image inputs, the number of pixels is really high. So if the relevant information can be obtained with reduced pixels without loss of important data, it will be of great use. In this work, seven features are considered such as contrast, correlation, entropy, energy or uniformity, homogeneity, mean and standard deviation. The chosen parameters are briefly described and computed in this work using standard formulae. The GLCM Matrix, a mathematical representation of texture patters in an image, is computed here. It is widely used in computer vision to characterize the spatial relationships between pixels with the same gray-level values in an image. The features so computed are given as input to the classifier and the class of the input data is predicted. The above mentioned features are chosen since they aid in image enhancement, image segmentation and texture analysis; that is, they provide quantitative information about the texture characteristics of the image. For instance, obtaining the contrast helps to perform image enhancement and segmentation. Evaluation of the entropy is greatly helpful in anomaly detection. In this work, this is pertinent because it can also help avoid incorrect decision making based on the captured images. Computing the mean and standard deviation is useful in image enhancement. Extracting contrast values are greatly useful for image segmentation and object detection. The features also help to perform the image quality assessment and, to some extent, noise reduction as well.

C. Classifiers

Classification is basically a supervised learning approach in machine learning. In this work, the following classifiers are used. In order to classify new observations, the computer programs will learn from the input data and use these learning to predict the output. This classification can be for binary (two class) classification or for multi-class classification. Various machine learning classifiers available are Naive Bayes classifier, Nearest Neighbour, Decision tree, Neural network, Support vector machines, Random forest, Logistic regression etc., and the common application of these classifiers are for document classification, speech recognition, handwriting recognition etc. Among these classifiers Support vector machines(SVM) and K nearest neighbour(KNN) are used here for classifying medicinal plants.

1) SVM classifier: Support vector machine algorithm is a machine learning algorithm. It can be used for classification as well for regression related problems. In this work we used it as a supervised classifier, where the linear discriminant function be given by the equation,

$$H(X) = W^T * X + C \tag{1}$$

2) KNN classifier: KNN is a machine learning algorithm that can be used either for classification problems or for regression problems when it is supervised. The main attribute which distinguishes classification and regression problem is that regression problem will be continuous and classification will be of discrete values. KNN is used as a supervised classifier in this work. Using KNN, the labels will be predicted based on the nearest neighbors and the image is classified. If the value of k is 3, the three nearest neighbors from the test data is calculated using the Euclidean distance. If two points belong to a given class C1 and the other neighbor is in class C2, then the majority number belongs to class C1 so the test data T is predicted to be in C1 by the KNN algorithm. It will produce better accuracy for higher values of k compared to that with least values of k. According to the number of classes and samples the decision of the k value can be chosen.

IV. IP Block

Intellectual Property (IP) block is a pre-configured logic or block of logic which can be incorporated in the design. It is reused whenever the functionality is required. In Vivado, there are inbuilt IP blocks which can be incorporated in the design. Multiplier, adder, BRAMs, etc are some commonly used IP blocks. The data width, depth, latency, etc can be adjusted in the configuration of IPs according to the requirement. In addition, if the design needs a particular function to be called several times then a custom IP can be created for such logic. In this work, a custom IP for an optimised supervised K nearest neighbour classifier is created and the functionality of the IP is verified. The following subsections highlight the how the custom IP is implemented in this work.

A. IP Creation

An IP block refers to a pre-designed and pre-verified block of digital logic or functionality that can be integrated into an FPGA project. It's a valuable tool for increasing productivity, reducing development time, and ensuring the reliability of FPGA-based systems. The custom IP creation requires certain steps to follow, these are given below:

- Creating the design
- Packaging the design IP
- Integrate system design and custom IP

1) Creating the design: Initially a project has to be created in Vivado design suite and design is created in it. Initially the simulation is performed using a test bench, so that the functional logic can be verified after which, the synthesis of the code is done. This is done so that our design will be converted into gate level representation. If the synthesis fails, then this design can't be implemented on the FPGA. After synthesis the implementation is done on xc7z020clg484-1 FPGA. If implementation is successful then the bitstreams have to be generated by giving the input-output pin descriptions. Bitstream generation is mandatory in IP generation for hardware implementation.

2) Packaging the design IP: In this step the packaging of our verilog code is done. As an interface the advanced extensible Interface (AXI) is used. It is a parallel high performance, multi slave, multi master, synchronous interface and is mainly used for on-chip communication purposes. In this work, a slave mode interface AXI lite interface is used. The data width is of 32 bit and memory of 64 bytes. Around 12 slave registers are used for storing the input and output parameters of the design. The procedure is to choose the option, create and package the new IP and then a new project will open with two files of AXI slave sub-modules. Now we will add our design and in the

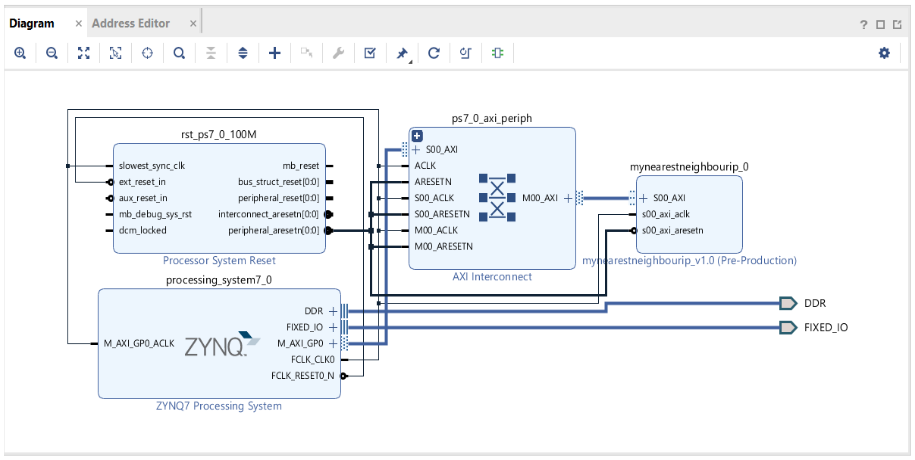


Fig. 2. Block design

AXI sub-module we will instantiate our design module using the slave registers and AXI clock. Then check if the code is synthesisable or not. If synthesis is done then re-package the IP, thereby completing the process.

3) Integrate system design and Custom IP: In this final step we have to add the our custom IP and the Zynq processor IP in the block level device. The block design represented in Fig.2 is the block used in this proposed work. The procedure is to place all the blocks such as our custom IP, Zynq processing system block and then perform run connection automation as well as run block automation so that all the other blocks such as peripherals and rest blocks will be placed. The clock frequency considered in this work is 100 Mhz. After saving the design we have to validate it, if it is successful then move forward to other steps. Generate output products since the files can include the simulation targets, HDL and constraints. Further, an HDL wrapper is created, because the block designs cannot be synthesized directly. This creates a top level HDL which also has an instantiation template for the IP block design. Now the synthesis is done which is followed by the implementation process. If implementation is successful, then the pin allocation for inputs and outputs have to be specified and bitstreams are generated. The timing summary, area utilization and power consumption are noted after implementation is successful. Now we have to export the hardware along with bitstreams and then launch SDK. In SDK, the drivers will be there and the C code is written for accessing the PS part of the ZedBoard. Further, program the FPGA with the bitstreams. Using a serial terminal, say Tera term which is the terminal that we have considered here, can display the output when we launch the hardware debugger.

V. System Implementation

The leaf images of 10 species are taken from medicinal plants. The training set and test set images are separated. A total of 650 and 260 images respectively are there in trained and test set. The resizing of the image is also done. Then the color images are converted to gray images. The GLCM features are extracted from the gray images. About 7 selected features are considered for feature extraction as explained in Section. 3. These features are extracted from both the test set and training test. Using MATLAB software, the features are extracted and then the classification is performed using K nearest neighbour algorithm and support vector machine algorithm. When we input an image, the Manhattan distance will be calculated between this input image and the trained set. And the closest neighbours are observed. The classes of the majority of closest neighbours are noted and then the input images are grouped into that class. The number of closest neighbours (k) are not fixed initially, but the accuracy is noted for each k value. Following an earlier work of the authors [12], it was observed that for k value of 3 a maximum accuracy is obtained. Similarly, SVM algorithm based classification is also done based on the features extracted. But KNN algorithm optimized with k value as 3 is chosen for hardware implementation as it provides better accuracy. And since the data set is already labelled, the k nearest classifier in this proposed work is a supervised machine learning classifier.

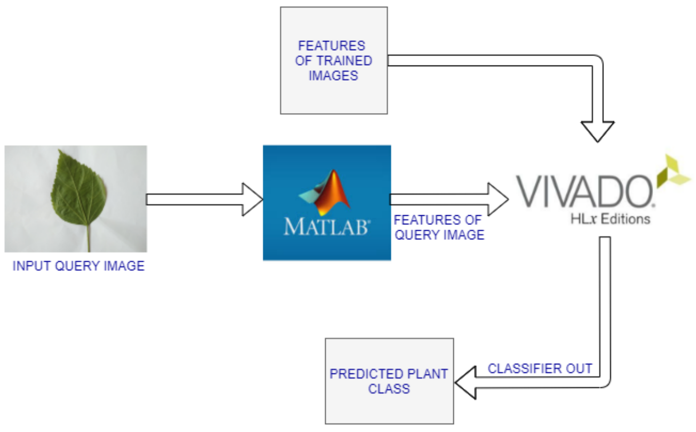


Fig. 3. Block diagram

The block diagram in Fig.3 shows the pictorial representation of the proposed work, in which the hardware implementation of the classifier is done. The feature extracted value for a particular input from MATLAB is given as an input to the Vivado. Here a new project is created and the verilog code for the classifier is written. The simulation is done for different inputs after which the synthesis and implementation of the design is checked. The implementation is done on FPGA xc7z020-clg484-1. The power utilization, area utilization and the timing summary are checked to ensure that it meets all the constraints. Finally, to calculate the accuracy of the classifier, the features in the trained set itself is given as input of the classifier and checked if it is giving correct output. Based on the result obtained, the confusion chart is plotted using MATLAB software and the accuracy of our classifier design is calculated.

To verify the functionality of the design, an IP creation is done for the design. For this, choose the option create and package IP and then specify the slave registers required. The data width will be of 32 bit and AXI lite slave interface is used. A new project with two source files of AXI interface is already available there and we have to add our design here. Overwrite the AXI files by instantiating our design module in it and including the slave registers. After making all these changes, the IP is repackaged. A new project is created in which a block design is created. Add the custom IP along with the Zynq processor IP which is already available in the IP catalog of Vivado. Then run connection automation and run block automation. The clock of PL is set as 100Mhz and the settings are saved. Validate the design and generate output products because this can include all the simulation targets, HDL and constraints. Then create HDL wrapper, this will relate all the physical pins described in the constraint file to the input-output pins of our design. Then run the synthesis and implementation and check the power, area and timings. If implementation is successful, then generate the bitstreams. The bitstream will contain the information to FPGA related to the programming. Once these bitstreams are generated, the design can be exported to SDK. In the SDK, the code for PS is written. The processing system will write data to the intellectual property (IP) and then read back the result. Then we have to connect the serial terminal Tera Term and configure the baud rate as 115200 bps, data as 8 bit, parity as none and finally stop bit as 1. Now program the FPGA (PL) with bitstream and also send the software application to the hardware (PS) part. Then the result from the PS will be displayed on the serial terminal.

VI. Results and Discussion

The features extracted from trained data set of 10 varieties of medicinal plant leaves using MATLAB are already stored in form of look up tables. And the feature extracted values from the test images are given as input. The distance of the given input from the stored features are calculated using Manhattan distance. The closest three neighbouring features to the input feature are taken. Identify the label of these neighbouring classes and classify our input to the label which is maximum among the neighbouring class labels.

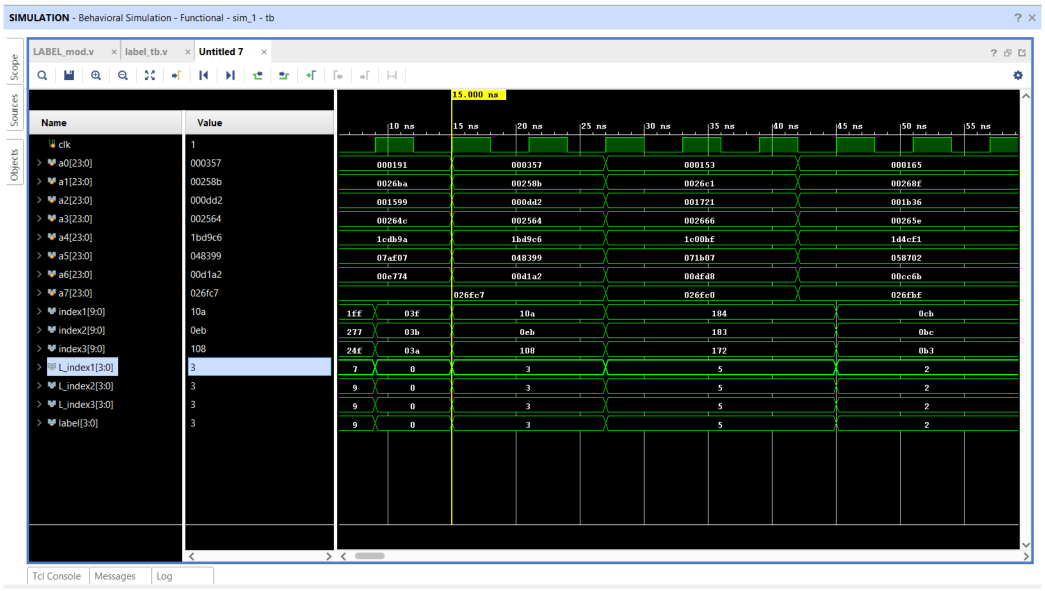
The feature extracted values from input image is the first eight signals in Fig.4. Since we optimized the k value as 3, the closest neighbors of the input is calculated using the Manhattan distance and it is indicated by the signals index1, index2 and index3. The corresponding class labels are also noted and indicated using the next three signals. Finally the maximum number of occurrence of the class labels is our predicted plant class and represented with the signal name label. And the plant names corresponding to the predicted plant labels are displayed in the TCL console window of vivado, as in Fig.6.

For the simulation, we considered 10 plant species such as Carissa, Plectranthus, Clerodandrum ,Maloora, Psidium guajava, Ocimum, Hibiscus, Samadera, Azadirachta and Morus and verified the simulation result. In order to find the accuracy of the classifier and compare it with the accuracy obtained in MATLAB, the feature extracted values of the images belonging to the data set is given as input to the verilog design and checked the output class. This process is done for all the images in the data set and noted the output class. And then with the help of MATLAB a confusion chart is plotted. Confusion chart is the plot between the true class and the predicted classes, the diagonal elements will give the sum of true positive and true negative values which are the correct predictions. This confusion chart as well as the accuracy calculated from this confusion chart is shown in figure Fig.7.

The accuracy value is calculated using the equation using the true positive, true negative, false positive and false negative values and it is about 88.3077 %. And when the accuracy was calculated for the KNN classifier in MATLAB it was about 100 %. So a comparison is made between the accuracy values produced by MATLAB and the design in Vivado. The synthesis and implementation of the design is done on xc7z020clg484-1 FPGA. And the power report, area or utilization report and timing reports are observed. The power report for the design is represented in Fig5. The power consumed by the clock, signals, input & output and logic is also represented in the figure. The device static power is 0.002 W and device dynamic power is 0.104 W. So the total on chip power is 0.106 W which is low power. The utilization report includes the components utilized for the design such as flip flops, look up tables, BRAMS, IOs etc. The utilization in percentage will indicate the ratio of used resources for the design to the available resources for the FPGA and multiplied by 100. This report is shown in Fig.8. The design is implemented on xc7z020clg484-1 at a clock frequency of 100 MHz and it is observed that all the timing constraints such as worst negative slack, worst hold slack and worst pulse width slack are greater than or equal to zero, implying that all the constraints are met.

TABLE I
UTILIZATION PERCENTAGE

Component	Utilization(l)	Utilization(%)
LUTs	32850	61.75
FF	1101	1.03
IO	167	83.50
BUFGCTRL 1.1	1	3.13

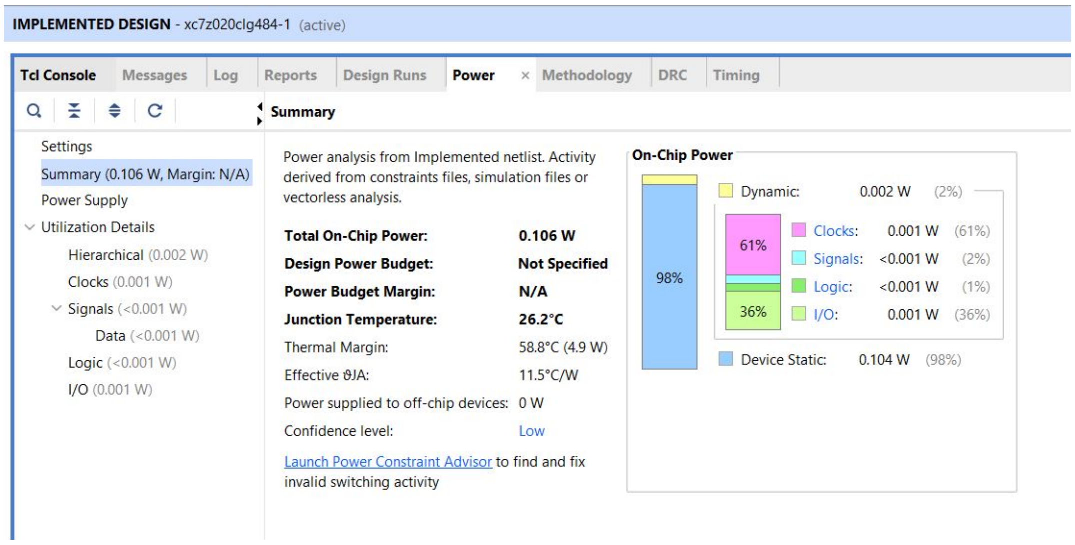


SIMULATION - Behavioral Simulation - Functional - sim_1 - tb

LABEL_mod.v | label_tb.v | Untitled 7

Name	Value
clk	1
a0[23:0]	000357
a1[23:0]	00258b
a2[23:0]	000dd2
a3[23:0]	002564
a4[23:0]	1bd9c6
a5[23:0]	048399
a6[23:0]	00d1a2
a7[23:0]	026fc7
index1[9:0]	10a
index2[9:0]	0eb
index3[9:0]	108
L_index1[3:0]	3
L_index2[3:0]	3
L_index3[3:0]	3
label[3:0]	3

Fig. 4. Simulation waveform



IMPLEMENTED DESIGN - xc7z020clg484-1 (active)

Tcl Console | Messages | Log | Reports | Design Runs | Power | Methodology | DRC | Timing

Summary

Settings
Summary (0.106 W, Margin: N/A)
Power Supply
Utilization Details
Hierarchical (0.002 W)
Clocks (0.001 W)
Signals (<0.001 W)
Data (<0.001 W)
Logic (<0.001 W)
I/O (0.001 W)

Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

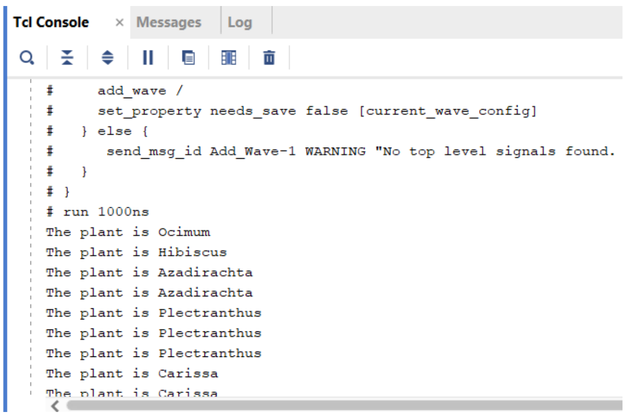
Total On-Chip Power:	**0.106 W**
Design Power Budget:	**Not Specified**
Power Budget Margin:	**N/A**
Junction Temperature:	**26.2°C**
Thermal Margin:	58.8°C (4.9 W)
Effective ϑJA:	11.5°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

Dynamic:	0.002 W	(2%)
Clocks:	0.001 W	(61%)
Signals:	<0.001 W	(2%)
Logic:	<0.001 W	(1%)
I/O:	0.001 W	(36%)
Device Static:	0.104 W	(98%)

Fig. 5. Power report



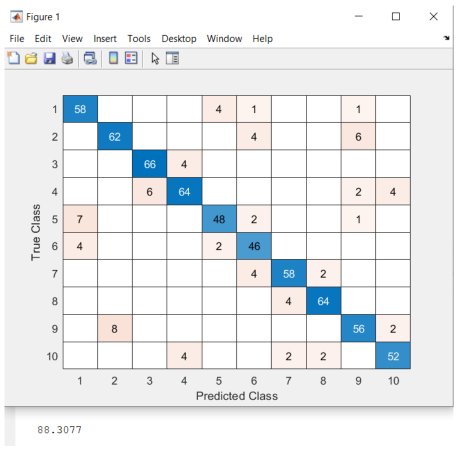
Tcl Console | Messages | Log

```
#     add_wave /
#     set_property needs_save false [current_wave_config]
#   } else {
#      send_msg_id Add_Wave-1 WARNING "No top level signals found.
#   }
# }
# run 1000ns
The plant is Ocimum
The plant is Hibiscus
The plant is Azadirachta
The plant is Azadirachta
The plant is Plectranthus
The plant is Plectranthus
The plant is Plectranthus
The plant is Carissa
The plant is Carissa
```

Fig. 6. Predicted plant names

The design is implemented on xc7z020clg484-1 at a clock frequency of 100 Mhz and it is observed that all the timing constraints such as worst negative slack, worst hold slack and worst pulse width slack are greater than or equal to zero, which means all the constraints are met.

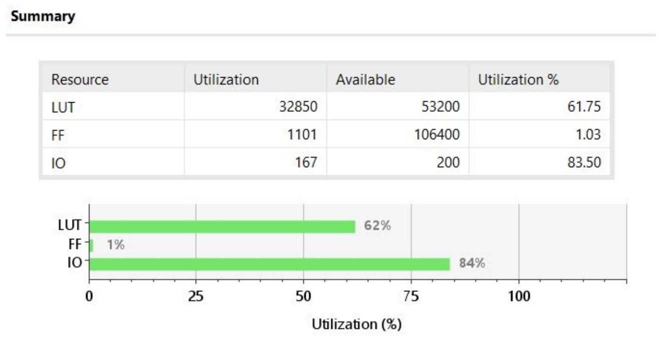
The design is used to create an IP using the AXI lite interface Along with the Zynq processing system, this IP will build our new design. The processing system will write data to the IP and the result is read. The RTL for the IP design is shown in Fig.9. The bitstreams are generated for the design, the hardware and bitstreams are exported and the SDK is launched. Then the application project is created and the C code is written where the inputs are provided for the processing system. The output value is read in the serial terminal. The experimental setup for this is shown in Fig.10.



True Class \ Predicted Class	1	2	3	4	5	6	7	8	9	10
1	58				4	1			1	
2		62				4			6	
3			66	4						
4			6	64					2	4
5	7				48	2			1	
6	4				2	46				
7						4	58	2		
8							4	64		
9		8							56	2
10				4			2	2		52

```
88.3077
```

Fig. 7. Confusion Chart for KNN classifier in Vivado


Summary
LUT
FF
IO
62%
1%
84%
0
25
50
75
100
Utilization (%)

Resource	Utilization	Available	Utilization %
LUT	32850	53200	61.75
FF	1101	106400	1.03
IO	167	200	83.50

Fig. 8. Utilization report

The Tera Term serial terminal with the results from the PS is shown in Fig.11. The first three classes of plants are considered and their classification for the input features is given in the figure. The label for the first, second and third group of plants is fixed as 0, 1 and 2 respectively. The functional verification on the hardware (ZedBoard) is also implemented successfully.

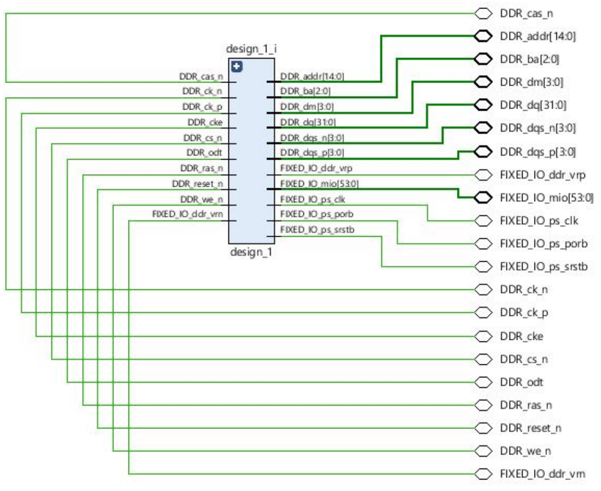


Fig. 9. RTL for IP Block Design

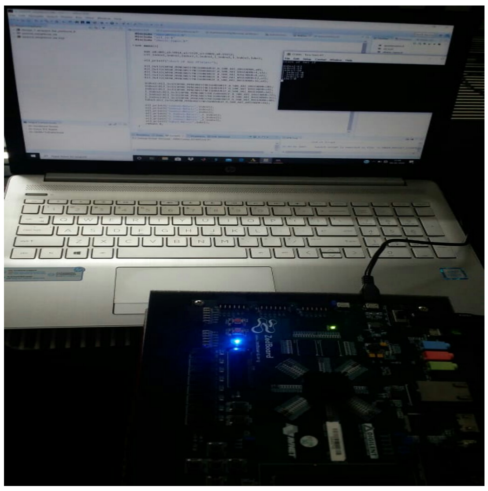
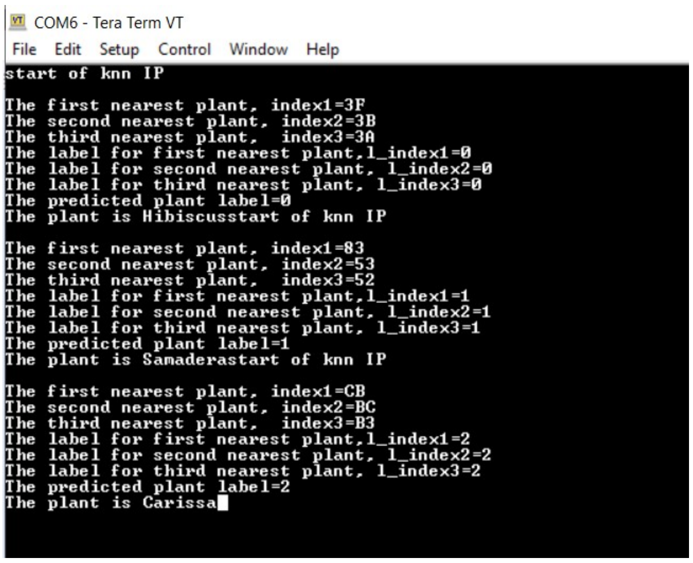


Fig. 10. Functional verification using ZedBoard



```
start of knn IP

The first nearest plant, index1=3F
The second nearest plant, index2=3B
The third nearest plant,  index3=3A
The label for first nearest plant,l_index1=0
The label for second nearest plant, l_index2=0
The label for third nearest plant, l_index3=0
The predicted plant label=0
The plant is Hibiscusstart of knn IP

The first nearest plant, index1=83
The second nearest plant, index2=53
The third nearest plant,  index3=52
The label for first nearest plant,l_index1=1
The label for second nearest plant, l_index2=1
The label for third nearest plant, l_index3=1
The predicted plant label=1
The plant is Samaderastart of knn IP

The first nearest plant, index1=CB
The second nearest plant, index2=BC
The third nearest plant,  index3=B3
The label for first nearest plant,l_index1=2
The label for second nearest plant, l_index2=2
The label for third nearest plant, l_index3=2
The predicted plant label=2
The plant is Carissa
```

Fig. 11. Plants detected from PS in Serial Terminal

VII. Conclusion

In this work, we successfully implemented a hardware realization for the medicinal plants classification. A detailed data set of the leaves of 10 medicinal plant species were constructed and features were extracted from the preprocessed images. They were classified with SVM and KNN classifiers. KNN algorithm was chosen for implementation as it showed accuracy as high as 100% compared to an accuracy of 93.23% for SVM. Then the hardware implementation for the optimised supervised k nearest neighbour classifier was then carried out on FPGA xc7z020-clg484-1. The accuracy of the classifier was obtained as 88.3 % when simulated in Vivado 2018.3. In order to perform a more comprehensive implementation, an IP block was created and its functionality was verified on ZedBoard for 3 different classes of plants. The simulation and hardware implementation results have been presented in this article.

Acknowledgment

We would like to express our heartfelt gratitude to Amrita School of Ayurveda, Amritapuri campus for granting access to visit and take images of several variety of medicinal plants at ”Amritavanam” - the herbal garden - which facilitated us with the data collection necessary for the successful completion of the work.

References

[1] Sandeep Kumar, V. T. E., Leaf feature based approach for automated identification of medicinal plants, 2014 International Conference on Communication and Signal Processing, Melmaruvathur, India, 2014, pp. 210-214, https://doi.org/10.1109/ICCSP.2014.6949830

[2] Sathwik, T., Yasaswini, R., Venkatesh, R., Gopal, A., Classification of selected medicinal plant leaves using texture analysis, 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT), Tiruchengode, India, 2013, pp. 1-6, https://doi.org/10.1109/ICCCNT.2013.6726793

[3] Păvăloiu, I.B., Ancuceanu, R., Enache, C.M., Vasilăţeanu, A., Important shape features for Romanian medicinal herb identification based on leaf image, 2017 E-Health and Bioengineering Conference (EHB), Sinaia, Romania, 2017, pp. 599-602, https://doi.org/10.1109/EHB.2017.7995495

[4] Yeni Herdiyeni, I. K., Fusion of local binary patterns feature for tropical medicinal plants identification, 2013 International Conference on Advanced Computer Science and Information Systems (ICACSIS), Sanur Bali, Indonesia, 2013, pp. 353-357, https://doi.org/10.1109/ICACSIS.2013.6761601

[5] Li, Z., Jin, J., Zhou, X., Feng, Z., K-nearest neighbor algorithm implementation on FPGA using high level synthesis, 2016, 600-602. https://doi.org/10.1109/ICSICT.2016.7998989

[6] Atitallah, M.B., Kachouri, R., Kammoun, M. Mnif, H., An efficient implementation of GLCM algorithm in FPGA, 2018 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC) (pp. 147-152). IEEE, 2018.

[7] Tian, M., Wang, X., Zhang, X., Yang, Z., Huang J., Chen, H., The implementation of a KNN classifier on FPGA with a parallel and pipelined architecture based on Predetermined Range Search, 2016 13th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT), Hangzhou, China, 2016, pp. 1491-1493, https://doi.org/10.1109/ICSICT.2016.7998779

[8] Venkataraman, M. N. D ., Computer vision based feature extraction of leaves for identification of medicinal values of plants, 2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), Chennai, India, 2016, pp. 1-5, https://doi.org/10.1109/ICCIC.2016.7919637

[9] Nidhis, A.D., Pardhu, C.N.V., Reddy, K.C., Deepa, K., Cluster based paddy leaf disease detection, classification and diagnosis in crop health monitoring unit, Lecture Notes in Computational Vision and Biomechanics, vol 31. Springer, Cham, 2019. https://doi.org/10.1007/978-3-030-04061-129

[10] Madhuri Bandara, L.R., Texture dominant approach for identifying ayurveda herbal species using flowers, 2019 Moratuwa Engineering Research Conference (MERCon), Moratuwa, Sri Lanka, 2019, pp. 117-122, https://doi.org/10.1109/MERCon.2019.8818944

[11] Saleem, G., Akhtar, M., Ahmed, N., Qureshi, W.S., Automated analysis of visual leaf shape features for plant classification, Computers and Electronics in Agriculture, Vol.157, 2019, pp. 270-280, https://doi.org/10.1016/j.compag.2018.12.038

[12] Raghukumar A.M., Narayanan, G., Comparison Of Machine Learning Algorithms For Detection Of Medicinal Plants, 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 2020, pp. 56-60.