Copyright © 2018 Tech Science Press CMES, vol.114, no.2, pp.209-220, 2018

An Ensemble Based Hand Vein Pattern Authentication System

M. Rajalakshmi[1, *], R. Rengaraj[2], Mukund Bharadwaj[3], Akshay Kumar[4], N. Naren Raju[5] and Mohammed Haris[6]

Abstract: Amongst several biometric traits, Vein pattern biometric has drawn much attention among researchers and diverse users. It gains its importance due to its difficulty in reproduction and inherent security advantages. Many research papers have dealt with the topic of new generation biometric solutions such as iris and vein biometrics. However, most implementations have been based on small datasets due to the difficulties in obtaining samples. In this paper, a deeper study has been conducted on previously suggested methods based on Convolutional Neural Networks (CNN) using a larger dataset. Also, modifications are suggested for implementation using ensemble methods. Ensembles were used to reduce training time and cost by training multiple weak classifiers instead of a single, strong classifier. Classifiers used were CNN, Random Forest and Logistic Regression. An inexpensive and robust data acquisition system was also developed for obtaining the dataset. The obtained result shows an improved accuracy of 96.77% using ensemble method instead of dealing with a single classifier.

Keywords: Convolutional Neural Networks, Random Forest, Logistic Regression, ensemble, biometrics, vein pattern.

1 Introduction

Personal biometric systems have become highly popular over the last decade. The advent of high power computing capabilities in low cost devices has enabled these systems to become widespread. Traditional identification systems have always come under scrutiny due to their inherent drawback of having a physical device that can be stolen (smart card, key), or a passphrase that can be forgotten (password, pin code). Biometric systems circumvent these issues by using specific parts of the body to identify a person.

Such biometric methods have both advantages and disadvantages. Most recently, vein pattern biometrics have been explored in detail. Vein identification methods are intrinsically more secure than other biometric systems since it is not visible to the naked eye and cannot be easily reproduced. Vein patterns are obtained with the help of NIR sensors which only work in the presence of real blood, making it hard to trick. Additionally, this system is non-contact, making it highly attractive.

[1] Department of Information Technology, SRM University, Chennai-603203

[2, 3, 4, 5, 6] Department of Electrical and Electronics Engineering, SSN College of Engineering, Chennai-603110

[*] Corresponding Author: M. Rajalakshmi. Email: rajalakshmi.mo@ktr.srmuniv.ac.in

The main focus of this paper is to expand on previous studies by increasing the number of users to identify, thereby increasing the size of the database. The aim was to identify the impact of the increased dataset on training time and accuracy of the model. This task was split into two components. The first was to build hardware capable of being lightweight and mobile at a low price and made with commonly available parts. The second was to train and run the model without using high processing performance (large neural nets typically require GPUs to train). General machine learning algorithms have varying requirements-for e.g. Random Forests can identify feature maps well but have very high memory and CPU requirements while Support Vector Machine(SVM) has lower hardware requirements but may not work well for all cases.

Many papers have worked on creating a model to identify people using vein patterns. These papers make use of different methods both in the preprocessing and the training stages. In Bradski [Bradski (2000)], image thresholding is performed to binarize the samples, before the processed images are passed through a 4 layer CNN, consisting of fused convolutional and sub-sampling layers. The model is trained to classify a total of 50 subjects. In Ng [Ng (2000)], Niblack algorithm is used to perform image thresholding on the dataset, followed by morphology thinning algorithm. The histogram intersection method is used here to measure the similarity between histograms. In Breiman [Breiman (2001)], the preprocessing stage entails methods such as rotation correction and Efficient Local Binary Pattern. Random Forests are employed to make predictions. The images are classified into 256 distinct classes.

The proposed work aims to combine multiple models together to form an ensemble, with minimal use of resources and computational power while ensuring appreciable scalability. The rest of the paper is organized as follows: Section 2 deals with data acquisition, Section 3 with preprocessing methods applied on the dataset. Section 4 elucidates the various algorithms used in the paper and Section 5 discusses the results. Finally, the conclusion and future work is discussed in Section 6.

2 Data acquisition

A low cost and portable data acquisition solution was developed for these applications shown in Fig. 1. The implementation was done using the Raspberry Pi (R-Pi) as core for faster development time, ease of use and hardware integration.

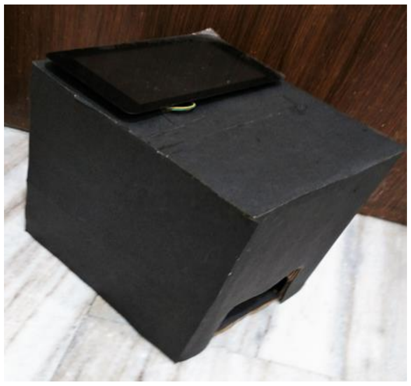

Figure 1: Image of data acquisition prototype

All the components required were housed in a portable box for easy transport. The components used were: Pi NoIR Camera, Near IR Light Source, Raspberry pi Touch Screen Display, Portable Power Source. The Pi NoIR Camera is a special version of the usual Pi Camera which does not have an Infra-Red (IR) filter. This makes it possible to capture the reflection of IR rays from the hand from which the vein patterns are obtained. However, regular light spectrum is also captured by the camera. This spectrum can be eliminated by either using a high pass filter in front of the camera lens or by removing all sources of visible light. The latter was achieved by placing the camera inside an enclosed dark box. A Near IR light source was provided by using a 4×4 NIR LED array around the camera. The wavelength of light used was 850 nm.

The area of interest here was the vein patterns from the back of the person's hand. The grid of near-infrared LEDs along with the infrared camera automatically makes the veins appear dark and the rest of the hand light, thereby providing a good contrast between the two. The camera properties such as exposure and white balance were tweaked to increase the contrast and obtain high clarity of vein patterns.

The Touchscreen display with resolution 800×480 is connected to the R-Pi via DSI port for display output and I2C pins to capture touch input. The R-Pi is capable of supplying power to the display through a dedicated Universal Serial Bus (USB) interface and hence, a separate power supply is not required. While a headless device would reduce costs and increase portability, real time visual output was deemed necessary for immediate feedback on the positioning and quality of the images. Touch capabilities were also decided on for ease of use of the device. A high density portable power bank was used as the power supply for the device. This was done to ensure that easy replacement on the field, while collecting data, is possible. An opening large enough to place a person's hand inside the box was cut at the bottom. The touchscreen display and the power bank were placed on the outside at the top of the box. For the purpose of the rest of the paper, the captured pictures were then transferred to a dedicated computer.

3 Image Pre-Processing

Preprocessing of input samples is a crucial step for increasing the accuracy and the performance of a Neural Network. OpenCV [Bradski (2000)] for python was used for this purpose. It is an open source computer vision library written in C and C++. It is one of the most widely used software modules for computer vision and image processing. OpenCV was chosen for this project for three reasons 1) It is easy to install 2) It is open source 3) It has a plethora of functions that can also be tweaked easily. The flow graph for image pre-processing and extraction of Region of Interest (ROI) is shown in Fig. 2. Each operation is described in detail below.

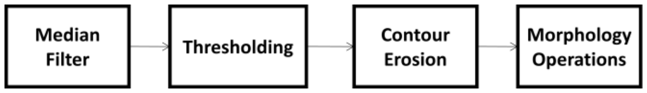

Figure 2: Flow graph for image pre-processing

3.1 Median blur

Blurring is an image processing operation that the main aim is to create a tool with the median of all the pixel intensity values in that neighborhood. Fig. 3 shows the hand vein image before and after applying median blur.

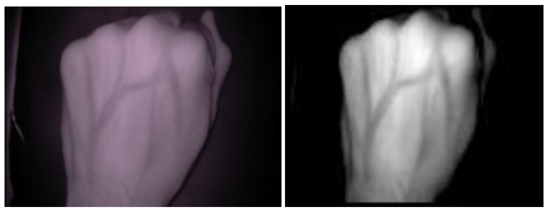
Figure 3: Before and after applying Median Blur

3.2 Adaptive mean thresholding

This process entails choosing a threshold value for pixel intensity, and setting the pixel intensity values to a background or a foreground value if it is above or below the threshold respectively. Instead of setting a common threshold for all the pixels in an image, adaptive thresholding dynamically sets the threshold for each pixel. Fig. 4 shows the hand vein image before and after applying adaptive mean thresholding.

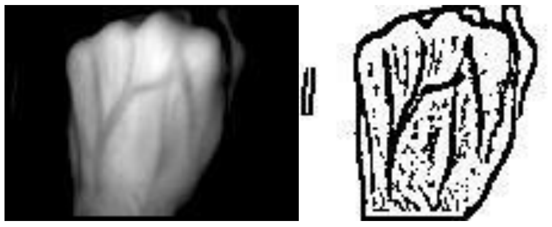
Figure 4: Before and after applying adaptive mean thresholding

3.3 Contour

A contour is a curve joining all the continuous points (along the boundary), having same color or intensity. In the proposed work, OpenCV's boundingRect function is used to approximate a rectangular contour around the region of interest. The resulting image after applying the function is shown in Fig. 5.

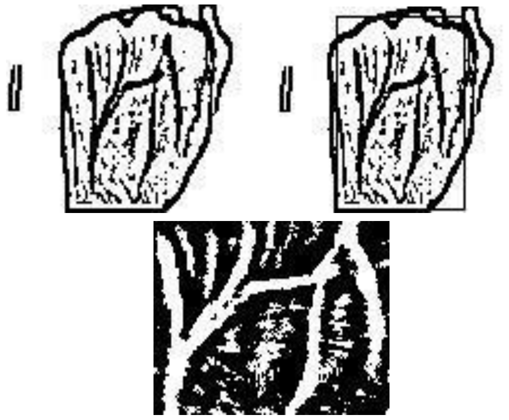
Figure 5: Result after applying the boundingRect function

3.4 Morphology operations

Morphological operations are performed on images to minimize some of the imperfections in them by applying a structural element on the image. Two basic morphological operations are erosion and dilation. Erosion is one of the two basic operations in the area of mathematical morphology. The basic effect of the operator on a binary image is to erode away the boundaries of regions of foreground pixels. Thus, areas of foreground pixels shrink in size, and holes within those areas become larger.

In addition to erosion, dilation is one of the basic operations in the area of mathematical morphology. The basic effect of the operator on a binary image is to gradually enlarge the boundaries of regions of foreground pixels. Thus, areas of foreground pixels grow in size while holes within those regions become smaller. The morphological operations done are shown in Fig. 6.

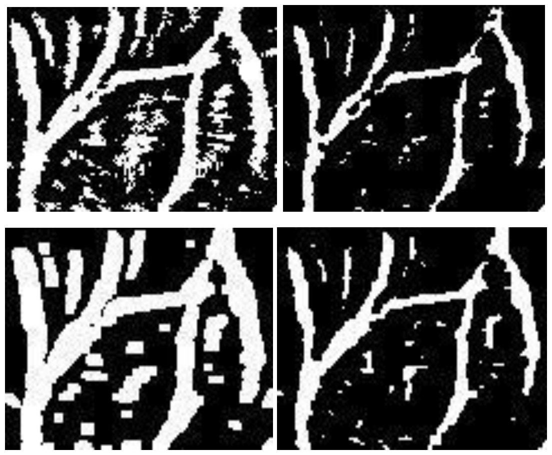

Figure 6: On applying morphological operations. From top left: a) image before any operation, b) after first erosion operation, c) after dilation, d) after second erosion operation (final image)

4 Various algorithms used

Initially various well-known architectures were applied directly with negligible modification. Various experiments were conducted for the purpose of optimization and improvement of the base model. Only methods that produced a significant change in accuracy have been explained in detail.

4.1 Convolutional Neural Networks (CNN)

The Convolutional Neural Network Model was chosen as the baseline as it has shown promising results in several classification tasks like handwritten digit recognition: LeCun et al. [LeCun, Jackel and Bottou (1995)], and face recognition: Hu et al. [Hu, Yang, Yi et al. (2015)] compared to other standard methods. It has also been proven to work for vein biometrics. The architecture of LeNet-5 has been looked at in detail and this was used as our standard of reference in order to determine a model's classification performance. Experiments were conducted on various attributes of the model in order to obtain an economic model based on computational efficiency and compromise between factors affecting accuracy. The weights were initialized using a uniform Glorot initialization. This is done in order to minimize standard deviation to avoid immediate saturation of the neurons as demonstrated by Glorot et al. [Glorot and Bengio (2010)].

The typical size of the sub-sampling window is 5×5 for small images. For the purpose of experimentation several other sizes of the local receptive field were used (3×3, 7×7), but produced negligible variation in accuracy. Thus, the size of the local receptive field was chosen to be 5×5, as used in the standard architecture of the LeNet-5. The number of feature maps used in each layer was determined in such a way that the amount of computation required for each layer is uniform. The number of feature maps was varied but produced negligible difference in accuracy as shown in Fig. 7 and 8.

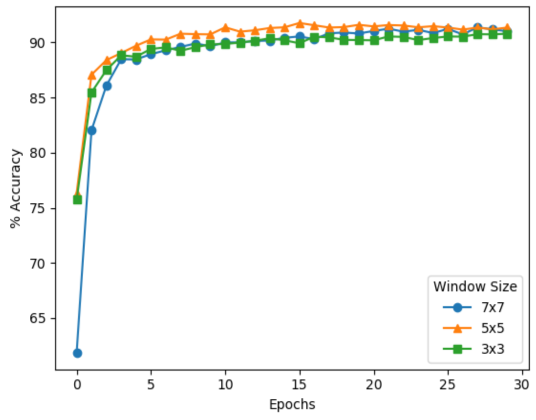

Figure 7: Variation of accuracy with size of local receptive field. 6 feature maps in first layer and 32 feature maps in second layer

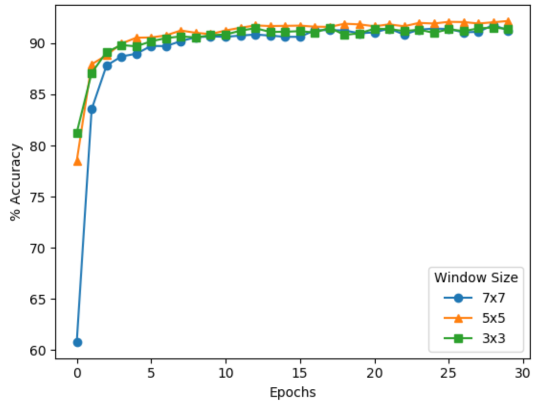

Figure 8: Variation of accuracy with size of local receptive field. 16 feature maps in first layer and 64 feature maps in second layer

A 2×2 pooling layer was made use of in the architecture. The classification problem was solved by minimization of the cross entropy cost function as given by below equation,

$$C = -\frac{1}{n}\sum_{x}\sum_{j}[y_j \ln a_j^L + (1-y_j)\ln(1-a_j^L)\ ]$$

The Rectified Linear Unit (ReLu) was chosen to be the activation function for all the neurons with the exception of the Softmax layer used for final classification. Adadelta algorithm was used to optimize variation in learning rate so as to reduce the occurrence of the dying ReLu problem, where the neurons may be pushed into inactive states such that no gradient flows backwards through the neuron and effectively decrease the model capacity. The method dynamically adapts over time using only first order information and has minimal computational overhead beyond vanilla stochastic gradient descent. The method requires no manual tuning of a learning rate and appears robust to noisy gradient information, different model architecture choices, various data modalities and selection of hyperparameters [Athiwaratkun and Kang (2015)].

4.1.1 CNN as feature extractor

The use of pre-trained Neural Networks as feature extractors such as in Ng [Ng (2000)] has been shown to increase classification accuracy. Logistic Regression was used to classify the extracted features from the above trained CNN. As dropout regularization methods were applied during training as specified before, stochastic behavior was disabled before extracting said features.

Logistic Regression is a classification algorithm that makes use of the logistic or the sigmoid function. The goal is to estimate a value such that the probability for a particular output given the input is as large as possible. The equation for determining the probability is given as follows,

$$P(x) = h_\theta = \frac{1}{1+\exp(-\theta^T x)} \equiv \sigma(\theta^T x)$$

$$P(x) = 1 - P(x) = 1 - h_\theta(x)$$

The goal is to minimize the cost function used so as to produce a decent approximation of the maximum probability. Determination of the parameters via maximum likelihood method and other useful properties like the derivative of the sigmoid function, that were made use of in the experimentation process has been explained in detail by Andrew Ng in his Lecture Notes Breiman [Breiman (2001)]. The feature maps obtained from the selected layer is as shown in Fig. 9.

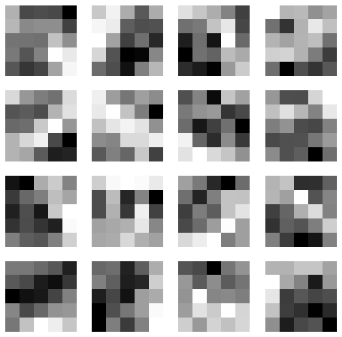

Figure 9: Feature maps obtained from layer 1 of CNN

Each feature map obtained for each training image was used to train the logistic regression model. During testing, the output probabilities were computed for each feature map. These outputs were averaged to obtain the final probabilities for any given image. The final user was selected by selecting the highest probability using a greedy model.

4.2 Random forest

Random Forest is a machine learning algorithm that forms an ensemble of decision trees. Multiple subsets of the training data are first made. Then, decision trees are constructed for each of these subsets. The testing data, for which the prediction has to be made, is then passed into each of these decision trees, and the outputs from each of the trees are collected. The final prediction is the class that gets the most number of "votes" from the decision trees. The algorithm was first put forth in Dietterich [Dietterich (2000)]. Since it uses an ensemble of decision trees formed from different subsets of the training data, overfitting, as well as variance, is greatly reduced.

4.3 Bagging

Bagging traditionally means splitting the dataset into different sub-samples, building a machine learning model on each of them, and averaging the individual predictions. This is usually done to reduce variance and over-fitting.

4.4 Ensemble of networks

It has been well documented that an ensemble of machine learning networks can prove to be more effective than a single model. It must be noted that a necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual members is if the classifiers are accurate and diverse. An accurate classifier is one that has an error rate of better than random guessing on new values. Two classifiers are diverse if they make different errors on new data points [Bradski (2000)].

In this paper, an ensemble of the above mentioned classifiers has been experimented. Instead of using a voting based ensemble, a weighted average was used. The probabilities of every prediction from each model were used to calculate the final probabilities of the ensemble using weighted multipliers. Finally, the prediction was done by selecting the highest probability using a greedy model. Before the individual predictions were ensembled, each prediction was first normalized using L2 normalization. This was done to give more preference to highly confident predictions, regardless of the final weight assigned to the model.

5 Results

Variations in several hyper parameters were experimented upon and the effects they posed on the variation in accuracy have been discussed in this section. 10 samples from 403 hand labels was used for the experiment. Training, validation and testing sets were obtained by a 5:3:2 split of the original dataset. All training times mentioned are based on a regular Google Cloud Instance with 32 GB ram and 8 core processor running Debian. All models were implemented completely in python, using all 8 cores for training. The CNN model was built using Theano [Bradski (2000)] as its base. Sci-kit Learn [Theano

Development Team (2016)] was used for building the Random Forest and Logistic Regression models. The second CNN architecture proposed, with 16 feature maps in the first convolutional layer and 64 feature maps in the second layer was selected as the base model for comparison. Regularization was achieved using two dropout layers. The dropout rate was set to 0.5 in our model. The model was trained for a fixed number of epochs -100.

Bagging was done by initialising the logistic regression and random forest models with 10 different random states. The resulting 10 models were averaged and used for prediction. This increased the prediction accuracy by about 4%.

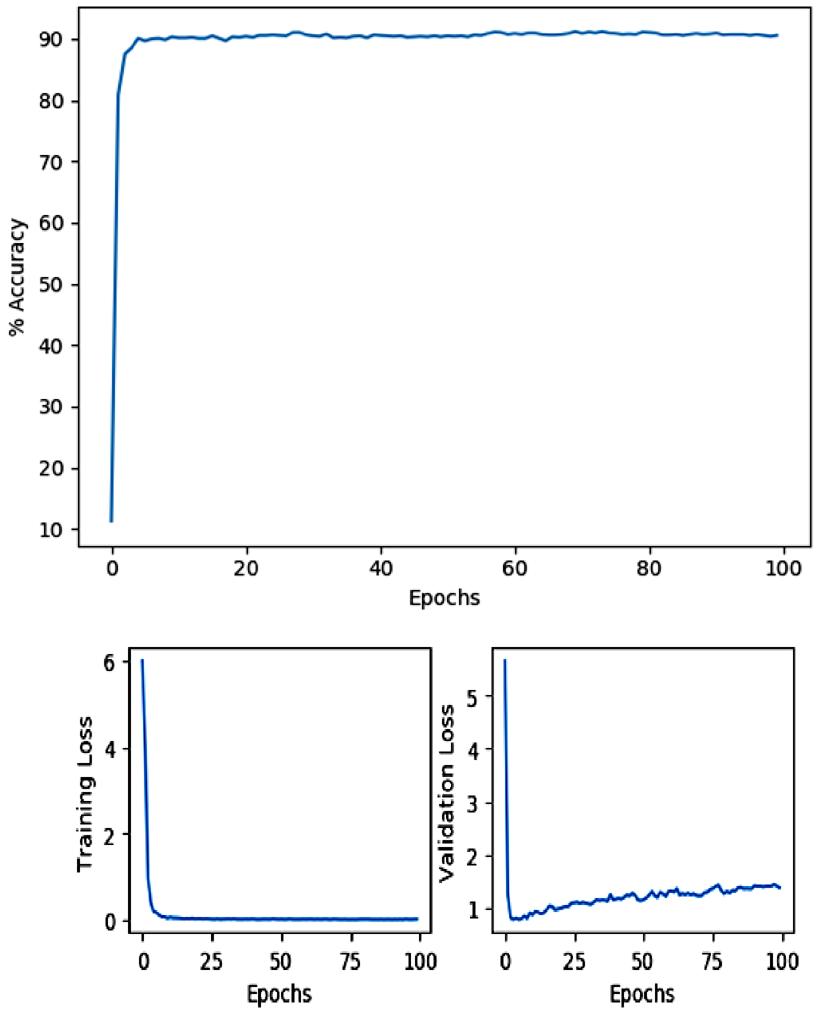

Figure 10: Plot showing variation in validation accuracy, training loss and validation loss

Feature maps for each label were also extracted from the trained CNN from the two convolutional layers as mentioned above. Finally, an ensemble of the three models (ensemble 1) was also generated as shown in Fig. 11.

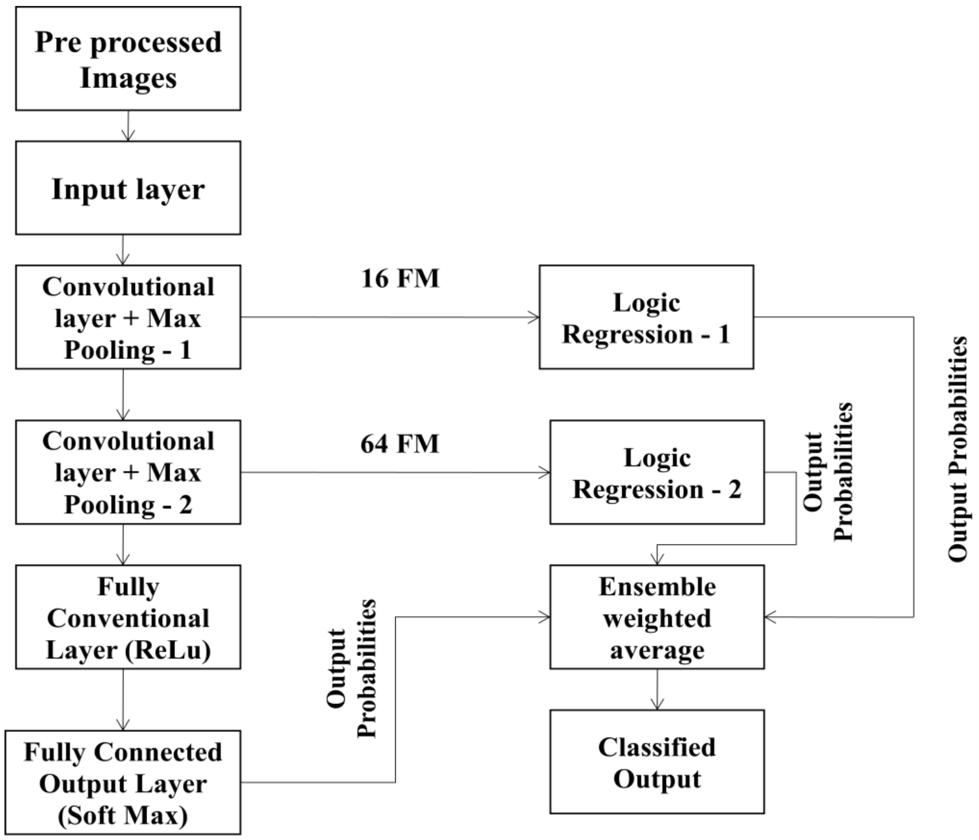
Pre processed Images
Input layer
Convolutional layer + Max Pooling - 1
16 FM
Logic Regression - 1
Output Probabilities
Convolutional layer + Max Pooling - 2
64 FM
Logic Regression - 2
Output Probabilities
Fully Conventional Layer (ReLu)
Output Probabilities
Ensemble weighted average
Fully Connected Output Layer (Soft Max)
Classified Output

Figure 11: Model of ensemble-1

The final accuracy obtained from each model is shown in Tab. 1. An accuracy of 86.48% was obtained from the pure CNN model. The logistic regression models trained on Layer 1 and Layer 2 of the CNN also showed similar or better accuracies. Both logistic regression models were trained using Scikit-Learn's SGD Classifier in order to train the model with batches and to enable multi-core training. L2 regularization was used while training to bring sparsity to the models.

An ensemble of the 3 models was also trained. Here the effect of ensemble is clearly seen and the accuracy of the final model obtained from the CNN is approximately 5% greater. In addition, a Random Forest model using 1500 trees was trained on the dataset.

The model was also tested against varying number of labels to check if higher number of classes caused any adverse changes in accuracy. As shown in Fig. 12, it can be seen that even with the drastic fall in CNN accuracy when tested with 200 classes, the final ensemble accuracy is fairly stable and does not dip below 95%. Similarly, although the accuracy for the linear regression model (LR2) falls to less than 85% when tested with all 403 samples, the final model shows an increase in accuracy. This is a good demonstration of the advantages of the proposed ensemble. Also, from Tab. 1, it is observed that ensemble 2 gives the highest accuracy among all the other models trained. However, the increase in accuracy is not much when compared to that of ensemble 1. Moreover, ensemble 2 takes a longer time to run and requires more system resources. The random forest model alone required more than 90% of memory on the system used. Hence, ensemble 1 is the model of choice for this paper.

Table 1: Classification accuracy and execution time for CNN, Logistic Regression and ensemble model 1

Model	Accuracy (%)	Execution Time (s)
CNN(100 Epochs)	86.48	2467
Logistic Regression-Layer 1 (LR1)	96.15	302
Logistic Regression-Layer 2 (LR2)	83.50	266
Ensemble 1: CNN, LR1, LR2	96.53	3035
Random Forest	92.68	269
Ensemble 2: CNN, LR1, LR2, Random Forest	96.77	3304

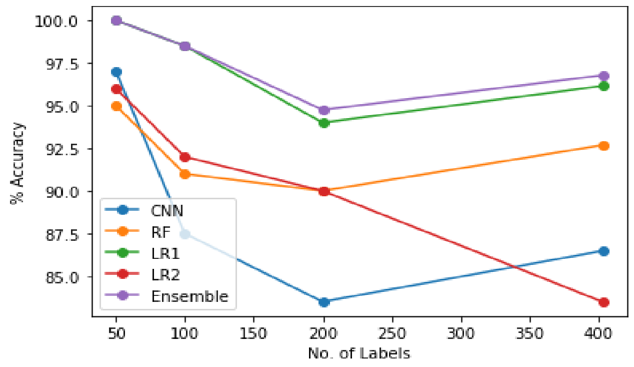

Figure 12: Variation in accuracy with number of labels

6 Conclusion and Further Discussion

Vein pattern biometric has been the subject of a great deal of attention due to its difficulty in reproduction and inherent security advantages. These solutions are generally based on image processing techniques, dissimilar to other biometrics such as fingerprint. Since biometric solutions tend to be very distributed, it becomes imperative that the solutions are capable of running on low cost devices with lower system requirements. In this paper, a low resource-intensive, ensemble based approach has been shown to improve the accuracy and rejection rate of the Convolutional Neural Network (CNN) model, with little additional training cost.

Further study can be done on improving the ensemble by training other machine learning algorithms or by using other well documented ensemble techniques such as variation in input data between models.

References

Athiwaratkun, B.; Kang, K. (2015): Feature representation in convolutional neural networks, arXiv:1507.02313v1 [cs.CV].

Bang, C. L.; Shan, J. Xie.; Dong, S. P. (2016): Finger vein recognition using optimal partitioning uniform rotation invariant LBP descriptor. *Journal of Electrical and Computer Engineering*, vol. 2016, pp. 1-10.

Bradski, G. (2000): The open CV library. *Doctor Dobbs Journal*, vol. 25, no. 11, pp. 384-386.

Breiman, L. (2001): Random forests. *Machine Learning*, vol. 45, no. 1, pp. 5-32.

Dietterich, T. G. (2000): Ensemble methods in machine learning. *Multiple Classifier Systems*, vol. 1857, pp. 1-15.

Glorot, X.; Bengio, Y. (2010): Understanding the difficulty of training deep feedforward neural networks. *Journal of Machine Learning Research*, vol. 9, pp. 249-256.

Hu, G.; Yang, Y.; Yi, D.; Kittler, J.; Christmas, W. et al. (2015): When face recognition meets with deep learning: an evaluation of convolutional neural networks for face recognition. *IEEE International Conference on Computer Vision Workshop*, pp. 384-392.

LeCun, Y.; Jackel, L. D.; Bottou, L.; Brunot, A.; Cortes, C. (1995*)*: Comparison of learning algorithms for handwritten digit recognition. *International Conference on Artificial Neural Networks*, pp. 53-60.

Liu, C.; Kim, Y. H. (2016): An efficient finger-vein extraction algorithm based on random forest regression with efficient local binary patterns. IEEE International Conference on Image Processing (ICIP), pp. 3141-3145.

Ng, A. (2000): CS229 Lecture notes.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B. et al. (2011): Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, vol. 12, pp. 2825-2830.

Radzi, S. A.; Khalil-Hani, M.; Bakhteri, R. (2016): Finger-vein biometric identification using convolutional neural network. *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 24, pp. 1863-1878.

Theano Development Team (2016): Theano: A python framework for fast computation of mathematical expressions, arXiv:1605.02688v1 [cs. SC].

Zeiler, M. D. (2012): ADADELTA: An adaptive learning rate method, arXiv:1212.5701v1 [cs. LG].