Exploring Ensemble Techniques With An Application In Hand Vein Biometrics

Mukund Bharadwaj1, Akshay Kumar1, N. Naren Raju1 and Mohammed Haris1

1Sri Sivasubramaniya Nadar College of Engineering, Chennai, TN 600xxx, India

Abstract

Many papers have dealt with the topic of new generation biometric solutions such as iris and vein biometrics. In particular, finger and hand vein biometrics have been the focus of a great number of studies. However, most implementations have been based on small datasets due to the difficulties in obtaining samples. A deeper study has been conducted on previously suggested methods based on Convolutional Neural Networks(CNN) using a larger dataset. Modifications have been 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.

***Index Terms* — *Convolutional Neural Networks, Random Forest, Logistic Regression, Ensemble, Biometrics, Vein Pattern Recognition***

# 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, pincode). 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.

Many papers have been published about the use of vein patterns from different parts of a person’s hand, eg: palm, finger, etc.[1,2,3,4] In specific, some papers have used Convoluted Neural Networks (CNN) to perform identification.

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 a 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 (most large neural nets require GPUs to train; general machine learning algorithms have varying requirements – for eg. Random Forests can identify feature maps well but have very high memory and CPU requirements while SVM has lower hardware requirements but may not work well for all cases)

# Data Acquisition

A low cost data acquisition solution was developed for this application. The implementation was done using the Raspberry Pi (R-Pi) as core for faster development time, ease of use and hardware integration.

All the components required were housed in a portable box for easy transport. The components used were :

1. Pi NoIR Camera
2. Near IR Light Source
3. Raspberry pi
4. Touch screen Display
5. 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 4x4 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.



*Figure.1. Image of Data Acquisition Prototype*

The Touchscreen display with resolution 800 x 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 usb 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.

# Image Pre-Processing

Preprocessing of input samples is a crucial step for increasing the accuracy and the performance of a Neural Network. Back Propagation Artificial Neural Network is the most commonly used algorithm for training Neural Networks. By tuning the various parameters of a Neural Network, its accuracy can be increased substantially. Many methods to do this have been put forth.

Later, it was found out that the techniques for Pre-processing the data also plays an important role in affecting the performance of MLP, where pre-processing data encourage the high accuracy and less computational cost associated to the learning phase [6].

OpenCV[12] 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 because:

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*

### 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 neighbourhood.

*Figure.2. Before and after applying Median Blur*

### Adaptive Mean Thresholding

As the name suggests, 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.

*Figure.3. Before and after applying Adaptive Mean Thresholding*

### Contour

A contour is a curve joining all the continuous points (along the boundary), having same color or intensity. Here, we used OpenCV’s **boundingRect** function, to approximate a rectangular contour around the region of interest.



*Figure.4. Result after applying the boundingRect function*

### Morphology Operations

Morphological operations are performed on images to minimise 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. 5

*Figure.5. On applying morphological operations. From top left, clockwise: a)Image before any operation, b)After first erosion operation, c)After dilation, d)After second erosion operation (Final Image)*

# Various algorithms used

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

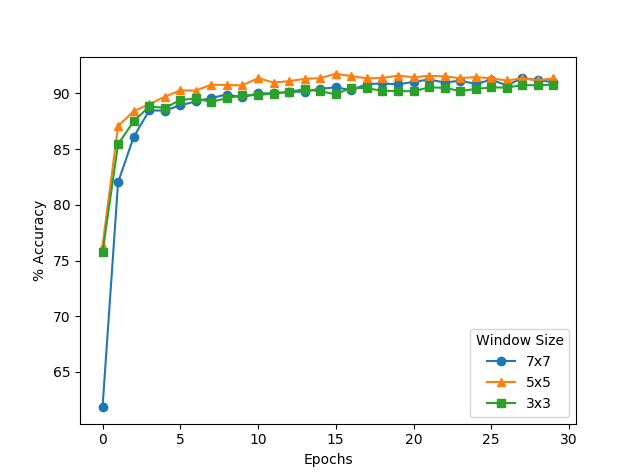
## Convolutional Neural Networks (CNN)

The convolutional neural Network model was chosen as the baseline as it has shown promising results in several classification tasks like handwritten digit recognition: Yann LeCun et. al. (1995) [6], and face recognition: Guosheng Hu et. al. (2015) [7] 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.

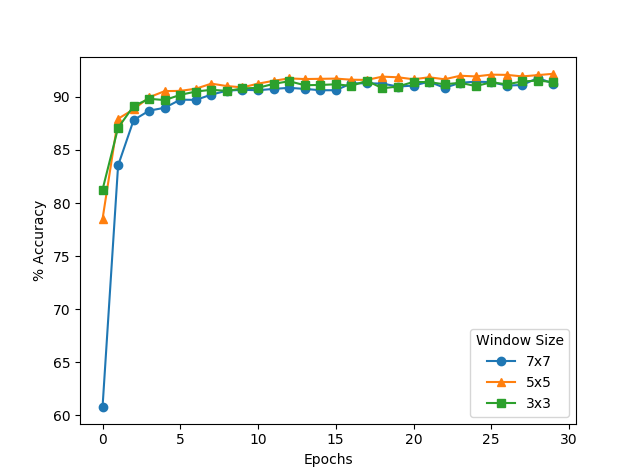
Experiments were conducted on various attributes of the model in order to obtain an economic model based on computational efficiency and compromise between factors affecting accuracy. The weights were initialized using a uniform Glorot initialization. This is done in order to minimize standard deviation to avoid immediate saturation of the neurons as demonstrated by Glorot and Bengio (2010) [8].

The typical size of the sub-sampling window is 5x5 for small images. For the purpose of experimentation several other sizes of the local receptive field were used (3x3, 7x7), but produced negligible variation in accuracy. Thus, the size of the local receptive field was chosen to be 5x5, 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.



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



*Figure.7. Variation of accuracy with size of local receptive field. 16 feature maps in first layer and 64 feature maps in second layer*

A 2x2 pooling layer was made use of in the architecture. The classification problem was solved by minimization of the cross entropy cost function.



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. The use of the Rectified Linear unit may lead to 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. This can be resolved by optimizing the variation in learning rate such that the occurrence of such a problem becomes rare. One such algorithm is Adadelta. 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 [9].

## CNN as Feature Extractor

The use of pre-trained Neural Networks as feature extractors such as in **[10]** 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.

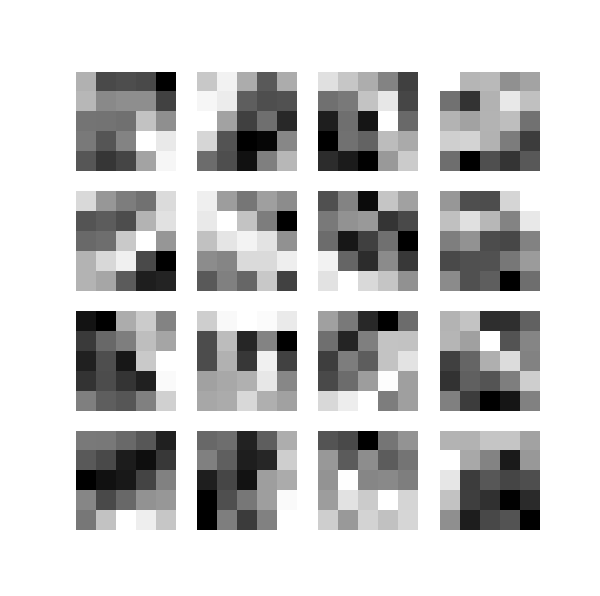
Logistic Regression is a classification algorithm that makes use of the logistic or the sigmoid function. The goal is to estimate a value of such that the probability for a particular output given the input is as large as possible. The equation for determining the probability is given as follows,





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 [11].

The feature maps obtained from the selected layer is represented as follows,



*Figure.8. 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.

## 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 [12].

Since it uses an ensemble of decision trees formed from different subsets of the training data, overfitting, as well as variance, is greatly reduced.

## 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 [13].

In this paper, an ensemble of the above mentioned classifiers has been experimented with. Instead of using a voting based ensemble, a weighted average was used. The probabilities of every prediction from each model was 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 ensembling the individual predictions, each prediction was first normalised using L2 normalisation. This was done to give more preference to highly confident predictions, regardless of the final weight assigned to the model.

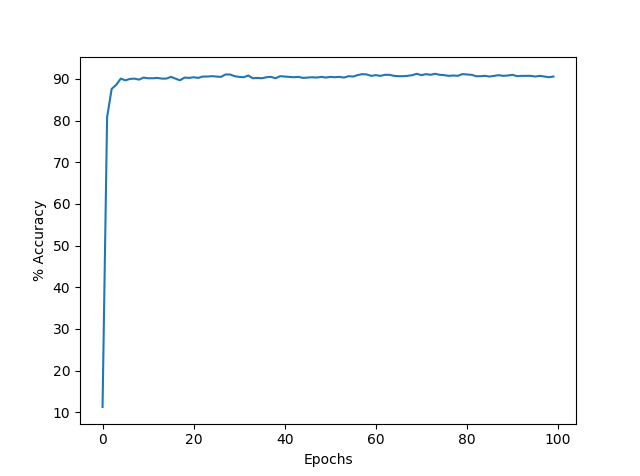
# Results

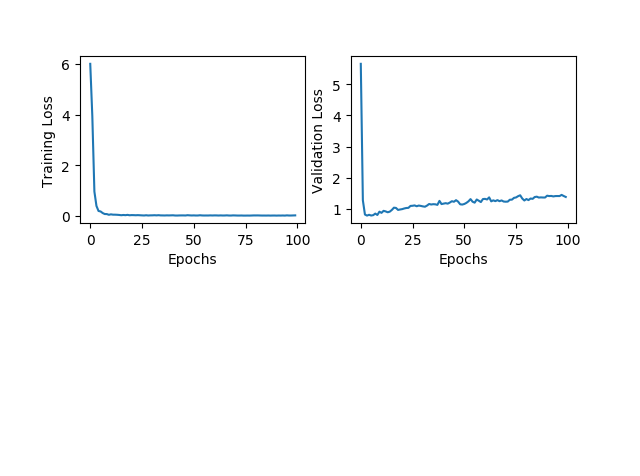
Variations in several hyperparameters were experimented upon and the effects they posed on the variation in accuracy have been discussed in this section.

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 a 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[13] as its base. Sci-kit Learn[14] 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 regularization method creates an ensemble of networks using a single training model. The dropout rate was set to 0.5 in our model. The model was trained for a fixed number of epochs - 100.

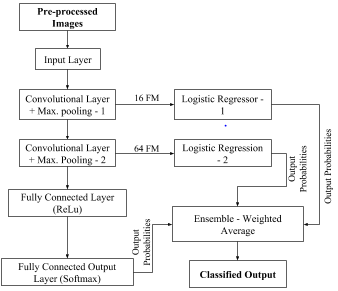
During the tuning process, it was also found that image augmentation played almost no difference to improving the model. While image augmentation is generally very good at expanding the database, it is thought that the limited augmentation done in this case, due to the restricted movement of the hand during data acquisition, along with the further pre-processing techniques done on the images causes the images to become very similar, leading to redundant samples.





*Figure.9. 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. 10.



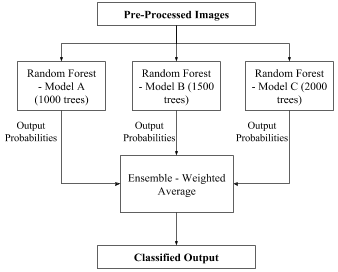
*Figure.10. Model of Ensemble - 1*

The final accuracy obtained from each model is shown in Table 1. An accuracy of 86.48% was obtained from the pure CNN model. Better accuracy was obtained from the logistic regression model trained on Layer 1 while the opposite was seen with Layer 2. 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. L1 regularization was used while training to bring sparsity to the models. An ensemble of the 3 models was found to achieve highest accuracy. Here the effect of ensemble is clearly seen and the accuracy of the final model obtained from the CNN is approximately 5% greater.

|  |  |  |
| --- | --- | --- |
| Model | Accuracy  (%) | Execution Time (s) |
| CNN (100 Epochs) | 86.48 | 2467.30 |
| Logistic Regression - Layer 1 (LR1) | 87.48 | 453.72 |
| Logistic Regression - Layer 2 (LR2) | 45.16 | 254.06 |
| Ensemble 1: CNN, LR1, LR2 | 91.32 | 3175.08 |

**Table 1. Classification accuracy and execution time for CNN, Logistic Regression and Ensemble Model 1**

Three different Random Forest models were trained with different number of trees. In this case, there was very little difference found between the models. The maximum number of trees were limited by the memory requirements of the algorithm. However, while looking at the detailed results, it was found that each model, while having similar accuracies, classified different labels correctly. The three models were therefore also ensembled (Ensemble 2) to get better accuracies and better confidence in the classified labels (Fig. 11).



*Figure.11. Model of Ensemble - 2*

The accuracy obtained from each model along with their training times are mentioned in Table 2

|  |  |  |
| --- | --- | --- |
| Model | Accuracy  (%) | Execution Time (s) |
| 2000 Trees (Model A) | 91.81 | 34.12 |
| 1500 Trees (Model B) | 91.68 | 25.76 |
| 1000 Trees (Model C) | 91.31 | 17.69 |
| Ensemble 2: Model A, Model B, Model C | 92.06 | 77.88 |

**Table 2. Classification accuracy and execution time for Random Forest and Ensemble Model 2**

Finally, both the ensembles were combined to give the final results. This gave an overall accuracy of 92.56%. The comparison of the three ensemble models are given in Table 3.

|  |  |  |
| --- | --- | --- |
| Model | Accuracy  (%) | Execution Time (s) |
| Ensemble 1 | 91.32 | 3175.08 |
| Ensemble 2 | 92.06 | 77.88 |
| Ensemble 3 | 92.56 | 3252.96 |

**Table 3. Classification accuracy and execution time of generated ensembles**

In biometrics, apart from accuracy, False Rejection Rate (FRR) and False Acceptance Rate (FAR) are important parameters to judge the efficacy of a model. Rejection is done by using a threshold for determining the minimum probability (confidence) for each prediction. It is preferred to have very low values of FAR to prevent improper authentication. This is done by setting a threshold which can distinguish successfully between correctly and incorrectly classified labels. However, increasing the threshold leads to higher FRR, which is not desired. Therefore, depending on the application, a balance should be found between FAR and FRR.

Using ensemble methods has the added advantage of reducing FRR for a given value of FAR. It can be seen from the table below that for a threshold of 0.58, there are 0 incorrectly classified labels while running the final model while achieving an accuracy of 87.34%. When applying a similar threshold to the pure CNN model, only a 72% accuracy is achieved for the same FAR. This is a 15% increase in accuracy from the original model. Table 4 shows the various values of accuracies for 0% FAR.

|  |  |  |
| --- | --- | --- |
| Model | Correct Classification  (%) | FRR  (%) |
| CNN (100 Epochs) | 72.20 | 27.80 |
| Logistic Regression - Layer 1 (LR1) | 55.21 | 44.79 |
| Logistic Regression - Layer 2 (LR2) | 5.71 | 94.29 |
| Ensemble 1 | 84.37 | 15.63 |
| Ensemble 2 | 81.76 | 18.24 |
| (Proposed Model) Ensemble 3: Ensemble 1, Ensemble 2 | 87.34 | 12.67 |

**Table 4. Percentage of correct classifications and FRR for different models**

The results obtained were compared against those of other similar papers and were tabulated as below:

|  |  |  |
| --- | --- | --- |
| Model | Number of Labels | Accuracy |
| CNN [1] | 81 | 99.38 |
| CNN [2] | 100 | 96 |
| LBPu2 [3] | 209 | 99.87 |
| RF [4] | 210 | 99.65 |
| Proposed Model | 403 | 92.56 |

**Table 5. Comparison of number of labels and accuracy across different models**

For the purpose of comparison, no threshold was set due to lack of corresponding information for the other models.

# 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 and random initialisation of models.

# References

1. Syafeeza Ahmad Radzi, Mohamed Khalil-Hani, Rabia Bakhteri, “Finger-vein biometric identification using convolutional neural network”, Turkish Journal of Electrical Engineering & Computer Sciences, 24, pp. 1863 – 1878, 2016
2. K. S. Itqan, A. R. Syafeeza , F. G. Gong, N. Mustafa, Y. C. Wong and M. M. Ibrahim, “User Identification System Based On Finger-vein Patterns Using Convolutional Neural Network”, ARPN Journal of Engineering and Applied Sciences, 11(5), pp. 3316-3319, 2016
3. B.C. Liu, S.J. Xie, D.S. Park, “Finger Vein Recognition Using Optimal Partitioning Uniform Rotation Invariant LBP Descriptor,” Journal of Electrical and Computer Engineering, vol. 2016, Article ID 7965936, 2016
4. Chenguang Liu, Yeong-Hwa Kim, “An Efficient Finger-vein Extraction Algorithm Based On Random Forest Regression With Efficient Local Binary Patterns”, 2016 IEEE International Conference on Image Processing (ICIP), 2016
5. N. M. Nawi, W. H. Atomi and M. Z. Rehman, "The Effect of Data Pre-processing on Optimized Training of Artificial Neural Networks," Procedia Technology, 11, pp. 33, 2013
6. Y. LeCun et al., “Comparison of learning algorithms for handwritten digit recognition”, in Fogelman, F. and Gallinari, P. (Eds), International Conference on Artificial Neural Networks, *53-60, EC2 & Cie, Paris, 1995*
7. When Face Recognition Meets with Deep Learning: an Evaluation of Convolutional Neural Networks for Face Recognition Guosheng Hu, Yongxin Yang, Dong Yi, [Josef Kittler](https://arxiv.org/find/cs/1/au:+Kittler_J/0/1/0/all/0/1), William Christmas, Stan Z. Li, Timothy Hospedales, [arXiv:1504.02351v1](https://arxiv.org/abs/1504.02351v1) [cs.CV]
8. X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks”. In Proc. AISTATS, volume 9, pp. 249–256, 2010.
9. M.D. Zeiler, “ADADELTA: An Adaptive Learning Rate Method”, arXiv:1212.5701v1 [cs.LG], 2012
10. B. Athiwaratkun, K. Kang, “Feature Representation In Convolutional Neural Networks,” arXiv:1507.02313v1 [cs.CV], 2015
11. CS229 Lecture notes written by Andrew Ng, direct link has been provided: cs229.stanford.edu/notes/cs229-notes1.pdf
12. L. Breiman, “Random forests”, Machine Learning, 45(1), pp. 5–32, 2001
13. T.G. Dietterich, “Ensemble Methods in Machine Learning”, Lecture Notes in Computer Science, vol 1857, 2000
14. G. Bradski,”The OpenCV Library”, Dr. Dobb's Journal: Software Tools for the Professional Programmer, 25(11), pp. 120-123, 2000
15. Theano Development Team, “Theano: A Python framework for fast computation of mathematical expressions”, arXiv:1605.02688v1 [cs.SC], 2016
16. Pedregosa et al., “Scikit-learn: Machine Learning in Python”, JMLR 12, pp. 2825-2830, 2011