

CS365

Project Report

Handwritten Hindi Numerals Recognition System

Submitted by:

**Akarshan Sarkar
Kritika Singh**

Project Mentor:

Prof. Amitabha Mukerjee

Abstract

In this project, we consider the problem of recognizing handwritten numerals using Machine Learning techniques. The first step is building the database using various image processing techniques like noise removal, elastic distortion, etc. We created a dataset of more than 200,000 Hindi numerals which is used to train and test our classifiers. Two neural network algorithms were used - backpropagation using gradient descent method and deep auto-encoders using the idea of Restricted Boltzmann machine. For each of the classifiers, after training the networks, their accuracy of classification was calculated and compared on the test set.

1 Introduction

Handwritten numeral recognition is a challenging problem of the field of Optical Character Recognition(OCR). The reason behind this problem being a difficult one is that many a times we see a large variation in the writing styles of different people. However, there is a great demand of such systems owing to their utility for various purposes like digitization of paper-based documents and automatic conversion of text as it is written on a Personal Digital Assistant(PDA). Recently, there has been an increase in app development for touch devices which use this technology. A hindi numeral recognition system can be used for making similar apps like a touch-based Hindi Calculator or a Hindi Sudoku Solver app.

2 Related work

Numerous people have used various combinations of image processing techniques, feature extraction methods and classification algorithms for the purpose of handwritten recognition. [1] after extracting histogram features from the images of numerals used backpropagation neural network algorithm for training their neural network.

3 Work done

Work done by us can be divided into two parts. A major part of our project was building the database of Hindi numerals as there does not exist any standard database like MNIST for Hindi numerals. The next part was identifying features and classification algorithms which give good results for our dataset.

3.1 Building the database

We built a database of more than twenty thousand numerals which is then expanded to more than two hundred thousand numerals . In the following section we describe the two stages involved in making the dataset - preprocessing and labeling

Table 1: Database statistics

Digit	Number of samples
0	2574
1	4306
2	2495
3	2466
4	1584
5	2816
6	1395
7	669
8	723
9	2941

Preprocessing

After removing non-uniformity in background color of the initial scanned data collection sheets, we converted the resulting images in RGB to grayscale. Noise removal is done on this grayscale image using an averaging filter so that each numeral can be identified as a single connected component. Images are then binarized by their global threshold calculated using matlab function `graythresh()`. This reduces the intraclass variance of black and white pixels. Bounding boxes are then found for each connected component. In addition to numerals, on the image we have some unwanted objects which could not be removed during noise removal. After observing the x width and y width of bounding boxes for various numerals, we set a threshold for these widths of bounding boxes and those boxes which lied in the range were to be counted as containing a numeral. The rest were mostly the stray objects talked about earlier and were discarded. The bounding boxes obtained are then cropped, resized to size 20x20 and finally centered on a zero matrix of size 28x28.

Database Expansion

A good dataset is one in which each class has a lot of variation within its training samples so that classifier trained on this dataset is insensitive to intraclass variability. For this reason, we used the two techniques of rotation and elastic distortion to expand the database.

- **Rotation** - We rotate each sample in the initial database by two randomly selected angles one in the range $+5^\circ$ to $+10^\circ$ and other in the range -5° to -10° [1].

Figure 1: Before and after rotation

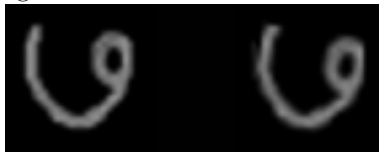
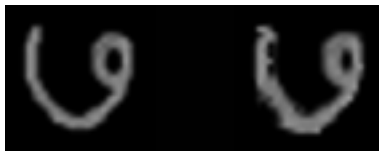


Table 2: Expanded Database statistics

Digit	Number of samples
0	25740
1	43060
2	24950
3	24660
4	15840
5	28160
6	13950
7	6690
8	7230
9	29410

- **Elastic Distortion**[4] - In this method, first random displacement fields (between -1 and $+1$) are generated with a uniform distribution. The displacement fields are then convolved with a Gaussian of standard deviation σ followed by normalization (by a norm of 1). For neither too high nor too low σ values, the displacement fields look like elastic deformation, where σ is the elasticity coefficient. Further, a scaling factor α which controls the elasticity of deformation is multiplied to this displacement field. This displacement field is then applied to the pixels in the original image to get new values for them.

Figure 2: Before and after elastic distortion



Labeling

The database obtained was too big to be labeled manually. Hence, k-means clustering algorithm was used to do unsupervised classification of the samples. Taking $k=10$, we obtained 10 clusters for each data file and assigned the correct label to each cluster. Few clusters had some misclassified samples which were manually removed and given the corrected label.

3.2 Feature extraction

- **Projection Histogram Features**[5] - The number of foreground pixels in four directions (horizontal, vertical, right diagonal and left diagonal) are calculated. These counts taken together constitute the feature vector of a sample.

- **Chaincode Histogram Features[2]** - Since the contour of a handwritten sample emarks the writing style, chaincode features give good results in digit recognition. From the contour representation of images obtained by canny-edge detector, we obtained chaincodes along this contour. These chain codes are given values 1, 2, .. 8 according to Freeman's direction code which are further quantized into one of the 4 values, viz. 1 or 5, 2 or 6, 3 or 7, 4 or 8 . Further, image is divided into 8 horizontal and vertical grids and local histogram of chain codes are calculated on these blocks to form the feature vector.

Codes: All the codes for preprocessing, rotation, elastic distortion, projection and chaincode histogram features were written by us.

3.3 Classification Algorithms

- **Backpropagation Neural Network Algorithm** - Backpropagation neural network with one hidden layer containing 30 nodes was used to train the classifier. Scaled conjugate gradient method was used for backpropagation. Validation set is used because if the accuracy over the training data set increases but stays the same or decreases on validation set (that has not been shown to the network before, or at least the network hasn't trained on it), means it is overfitting our neural network and we should stop training. The training goes on till the difference between mean square error of two consecutive epochs reaches a value less than a threshold.
- **Autoencoder networks[3]** - In this we use an adaptive, multilayer encoder network to transform the images into a low-dimensional code and a decoder network to recover the images. The two networks are trained together by minimizing the error between the two images. Backpropagating the error derivatives gradients are obtained first through the decoder network and then through the encoder network. The whole system is called an "autoencoder" .

For obtaining the initial weights of the autoencoder, images are pre-trained using a two-layer network called a "Restricted Boltzmann Machine"(RBM) obtaining a layer of binary features which become input for learning the next RBM. After pretraining 4 layers of feature detectors, model is "unfolded" to produce the autoencoder network. Then backpropagation fine-tune the weights.

3.4 Results

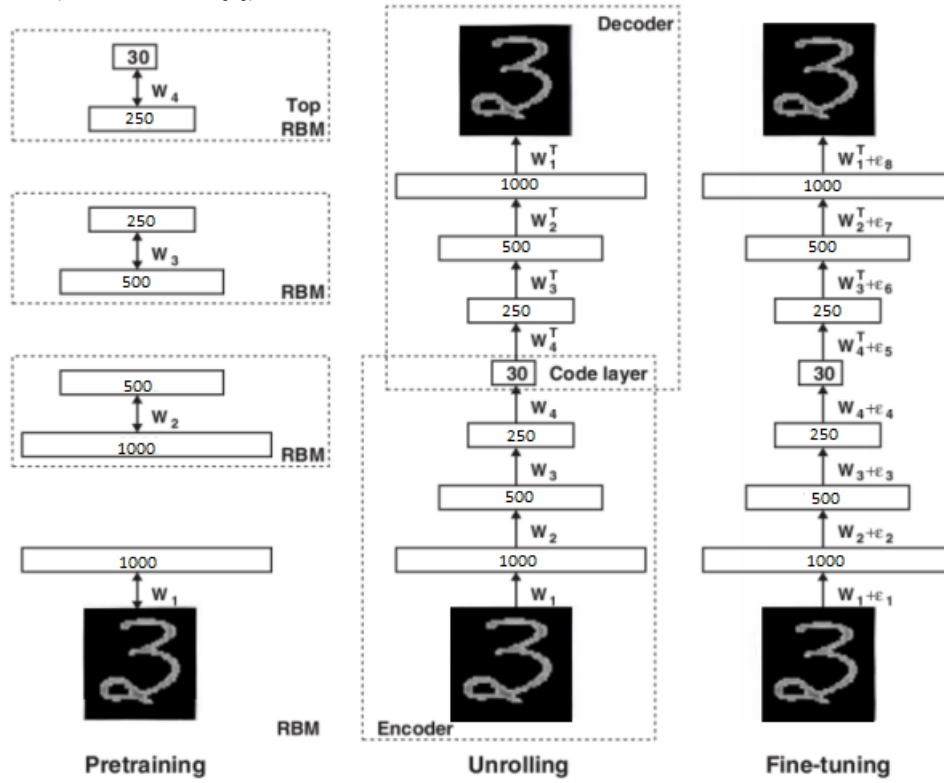
These results were obtained from the initial unexpanded dataset.

Backpropagation Neural Network Algorithm

[Used the MATLAB neural network toolbox]

The dataset obtained was randomly divided into 3 sets - 75% for training, 15%

Figure 3: Pretraining, unrolling, fine-tuning stages in a deep-autoencoder network.(Image source:[3])



for testing and the rest for validation. A confusion matrix, C , is also obtained where $C[i, j]$ gives the number of samples from the training set which have the class j and were classified by the network as i .

Using projection histogram features

Obtained an accuracy of 92.2% after training for 112 epochs

Test Confusion Matrix

1	620	9	1	0	4	2	0	4	13	0	94.9%
	9.0%	0.3%	0.0%	0.0%	0.1%	0.1%	0.0%	0.1%	0.4%	0.0%	5.1%
2	3	310	14	2	3	1	0	3	0	1	92.0%
	0.1%	9.5%	0.4%	0.1%	0.1%	0.0%	0.0%	0.1%	0.0%	0.0%	8.0%
3	1	12	319	6	2	13	0	1	3	0	89.4%
	0.0%	0.4%	9.8%	0.2%	0.1%	0.4%	0.0%	0.0%	0.1%	0.0%	10.6%
4	1	5	0	211	4	6	0	1	4	0	80.9%
	0.0%	0.2%	0.0%	6.5%	0.1%	0.2%	0.0%	0.0%	0.1%	0.0%	9.1%
5	2	8	1	9	400	3	3	0	2	0	93.5%
	0.1%	0.2%	0.0%	0.3%	2.3%	0.1%	0.1%	0.0%	0.1%	0.0%	6.5%
6	1	13	7	6	7	321	1	0	5	0	88.9%
	0.0%	0.4%	0.2%	0.2%	0.2%	9.8%	0.0%	0.0%	0.2%	0.0%	1.1%
7	1	1	1	0	1	0	52	1	0	2	88.1%
	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	1.6%	0.0%	0.0%	0.1%	1.9%
8	2	4	0	2	1	4	1	68	0	2	81.0%
	0.1%	0.1%	0.0%	0.1%	0.0%	0.1%	0.0%	2.1%	0.0%	0.1%	1.9%
9	18	3	0	4	2	3	0	4	315	2	89.7%
	0.6%	0.1%	0.0%	0.1%	0.1%	0.1%	0.0%	0.1%	9.7%	0.1%	10.3%
10	0	0	0	0	1	6	1	0	390	0	86.0%
	0.0%	0.0%	0.0%	0.0%	0.0%	0.2%	0.0%	0.0%	2.0%	0.0%	2.0%
	95.5%	4.9%	3.0%	7.9%	4.3%	30.7%	2.5%	1.9%	21.8%	8.2%	2.2%
	4.5%	5.1%	7.0%	2.1%	5.7%	9.3%	7.5%	8.1%	7.9%	1.8%	7.8%
	1	2	3	4	5	6	7	8	9	10	
											Target Class

Using chaincode histogram features

Obtained an accuracy of 92.7% after training for 217 epochs

Test Confusion Matrix

1	598	21	6	1	9	8	0	3	9	0	91.3%
	8.3%	0.6%	0.2%	0.0%	0.3%	0.2%	0.0%	0.1%	0.3%	0.0%	8.7%
2	16	331	6	1	5	1	1	1	0	0	91.4%
	0.5%	0.2%	0.2%	0.0%	0.2%	0.0%	0.0%	0.0%	0.0%	0.0%	8.6%
3	6	10	353	0	6	2	2	0	2	0	92.7%
	0.2%	0.3%	0.8%	0.0%	0.2%	0.1%	0.1%	0.0%	0.1%	0.0%	7.3%
4	1	1	1	214	6	3	1	0	2	0	93.4%
	0.0%	0.0%	0.0%	6.6%	0.2%	0.1%	0.0%	0.0%	0.1%	0.0%	6.6%
5	10	2	0	7	403	3	0	2	2	0	93.9%
	0.3%	0.1%	0.0%	0.2%	2.4%	0.1%	0.0%	0.1%	0.1%	0.0%	6.1%
6	4	7	4	1	3	319	1	1	3	1	92.7%
	0.1%	0.2%	0.1%	0.0%	0.1%	9.8%	0.0%	0.0%	0.1%	0.0%	7.3%
7	0	0	0	1	4	1	56	2	0	5	81.2%
	0.0%	0.0%	0.0%	0.0%	0.1%	0.0%	1.7%	0.1%	0.0%	0.2%	8.8%
8	5	0	1	3	2	0	4	75	2	0	81.5%
	0.2%	0.0%	0.0%	0.1%	0.1%	0.0%	0.1%	2.3%	0.1%	0.0%	8.5%
9	5	0	1	3	1	1	1	1	297	0	95.8%
	0.2%	0.0%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	9.1%	0.0%	4.2%
10	0	1	0	0	1	0	5	5	0	377	96.9%
	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.2%	0.2%	0.0%	1.6%	3.1%
	92.7%	8.7%	4.9%	2.6%	1.6%	4.4%	8.9%	3.3%	3.7%	8.4%	2.7%
	7.3%	1.3%	5.1%	7.4%	8.4%	5.6%	1.1%	6.7%	6.3%	1.6%	7.3%
	1	2	3	4	5	6	7	8	9	10	
											Target Class

Auto-encoder network [3]





[Code from <http://www.cs.toronto.edu/~hinton/MatlabForSciencePaper.html>]

After pretraining 4 layers with RBM each for 50 epochs followed by backpropagation for 70 epochs

Dataset	Accuracy	Total	Misclassified
Training	99.99%	19400	2
Test	98.2%	2500	45
0	99.3%	300	2
1	98.3%	300	5
2	97.3%	299	8
3	98.3%	301	5
4	98.3%	300	5
5	98.6%	300	4
6	99.3%	300	2
7	96%	50	2
8	90%	50	5
9	98.3%	300	5

4 Explaining the results

It was observed that most of the misclassification was due to similarities between some numbers, mainly 1,2 and 1,9 and different writing styles for the the same number like 9 and 8. Also the dataset for digit 7 and 8 were less in number which is also reflected in the accuracy of recognizing 7 and 8.

One as Two	Two as One	One as Nine	Eight as Seven
			

5 Conclusion

We developed the dataset for hindi numerals obtained from various people of variety of writing styles. We expanded this dataset using affine transformations and elastic distortion. We obtained projection and chain-code histograms on the dataset. Using backpropagation neural network classifier, we see that chain-code histograms are better feature-set than the projection histogram feature set. We also used deep autoencoder network classifier on the dataset which gave 98.2% accuracy which is because in this network the layers of the network are pretrained with RBM so that the weights of the layer are close to a good solution and gradient descent works well.

6 Future Work

In future we would like to use convolution neural networks as the classifier to compare and study the results. And also we would like to use some other features like zernike moments, and fourier descriptors .

References

- [1] Ujjwal Bhattacharya and Bidyut Baran Chaudhari. Handwritten numeral databases of indian scripts and multistage recognition of mixed numerals. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(444-457):3, 2009.
- [2] Ujjwal Bhattacharya, Malayappan Shridhar, and Swapan K. Parui. On recognition of handwritten bangla characters. In *ICVGIP*, pages 817–828, 2006.
- [3] G E Hinton and R R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–7, 2006.
- [4] Patrice Simard, David Steinkraus, and John C. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *ICDAR*, pages 958–962, 2003.
- [5] P.H. Wu. Handwritten character recognition. Master’s thesis, The School of Information Technology and Electrical Engineering, The University of Queensland.