


Sankhya: An Unbiased Benchmark for Bangla Handwritten Digits Recognition

Aminul Islam
Apurba Technologies
Dhaka, Bangladesh
aminul@apurbatech.com

Fuad Rahman
Apurba Technologies
Sunnyvale, CA, USA
fuad@apurbatech.com

AKM Shahariar Azad Rabby 
Apurba Technologies
Dhaka, Bangladesh
rabby@apurbatech.com

Abstract—The rise of artificial intelligence technology along with machine and deep learning are opening up almost limitless possibilities. In recent years, application-based researchers in machine learning and deep learning have started developing solutions for many practical problems. Handwriting recognition is one such area of interest. Bangla, being the seventh most spoken language in the world, is not an exception. However, unlike English, there have not been concerted formal attempts in building a benchmark in comparing the different approaches reported in the literature, mainly because of the lack of openly and freely available datasets and diversity of the approaches without formal comparative studies. In this research paper, we seek to rectify this gap. We have focused on benchmarking five robust algorithms: K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Random Forest (RF), Multi-layer Perceptron (MLP), Convolutional Neural Network (CNN), on all publicly available Bangla handwriting digits datasets, including Ekush, NumtaDB, CMARtdb, and BDRW. NumtaDB itself is a collection of five handwriting datasets. We have worked on fine-tuning these algorithms by finding the best possible hyper-parameters of these algorithms. It is our hope that Sankhya will work as a beginning point of an open and verifiable benchmarking process that we plan to repeat every two years for now on and set a standard for testing and validating newer and novel algorithms that will be reported in this area in the future. In addition, we have extensively compared our research with other states of the art research and our versions of these algorithms are now outperforming every reported result on these datasets.

All the datasets we used are open-sourced. In addition, we are making the .csv version of these datasets available in public GitHub. Of all the models we tested, the Sankhya CNN model performed the best for all these datasets, which we fine-tuned specifically for Bangla character recognition. We are making this CNN model available in public GitHub.

Index Terms—Ekush, NumtaDB, CMARtdb, BDRW, AI-Algorithm, Machine Learning, Benchmarking, Bangla Handwritten Digit Recognition, Sankhya.

I. INTRODUCTION

The world around us is changing dramatically. Artificial intelligence, Machine Learning, and Big Data Analytics are changing the way the world runs. Recognizing handwritten characters plays a key role in this change. The basic difference between one language to another language is their scripts. Bangla script is derived from Brahmi Script [1], also Bangla Script is somewhat influenced by Devanagari and Oriya scripts. In Bangla script, there are 50 basic characters, 10 digits, 10 modifiers, and more than 300 compound characters. For a long time, there were no standard databases

for Bengali handwritten digits. Recently a database, Ekush [2] was introduced with 3,670,018 unique characters. Ekush was collected based on age, an equal number of male and female writers and different geo-location, which is aimed at helping not only recognition tasks but also different forensic investigations. NumtaDB [3] with 85,596 handwritten Bengali digits, has been made open source by the efforts of Bengali.AI, a community trying to solve the absence of open-sourced datasets for Bengali Natural Language Processing/Computer Vision research by pulling together samples from six distinct data sources. CMARtdb dataset [4] is the most used handwritten dataset made by Jadavpur University. BDRW [5] collected data from real world Bangla script like books, papers, posters, etc. which made a robust dataset. Figure 1 shows an example of Bangla digits.

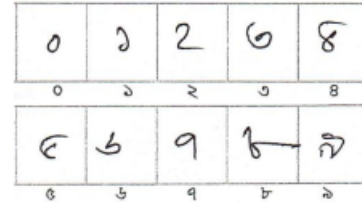


Fig. 1: All Bangla Digits

This has become the perfect opportunity to benchmark these databases against some well-known AI algorithms. In "Sankhya" we chose five algorithms, namely Convolutional Neural Network (CNN), Support Vector Machine (SVM), Multi-layer Perceptron (MLP), K-Nearest Neighbor (KNN) and Random Forest (RF) Classifier. These algorithms were implemented and tested against those databases. Table I shows the size of all datasets.

TABLE I: Number of samples in different dataset

Name of the Dataset	Number of samples
Numta	85,596
Ekush	30,688
CMARtdb	6,000
BDRW	1,393

II. LITERATURE REVIEW

For other languages like English and Chinese, lots of work has been done for handwritten digit recognition compared to

Bangla scripts. In the paper "Handwritten numeral databases of Indian scripts and multistage recognition" [6], the author used MLP classifiers on the mixed numerals. The matra, double line and vertical line on digit were identified using MLP.

In "Bangla Handwritten Digit Recognition Using Convolutional Neural Network" [7], the author used a convolutional neural network on three different datasets, named as ISI [6], CMARtdb [4], and BanglaLekha Isolated [8], to recognize the handwriting characters with Lenet [9] type convolutional network and achieved pretty good accuracy.

Another work "Character recognition based on non-linear multi-projection profiles measure" [10] whereby the author used the same algorithm in different languages including Roman, Japanese, Katakana, Bangla, etc. The author matched the non-linear multi-projection profile using dynamic programming.

In the paper "ShonkhaNet: A Dynamic Routing for Bangla Handwritten Digit Recognition Using Capsule Network" [11], the author used a complex network known as capsule network [12] to recognize the handwritten digits. Their validation accuracy was 98.90% in a combination of ISI, BanglaLekha-Isolated and CMATERdb datasets. The main feature of the capsule network is that it can work pretty well on overwritten characters.

III. RESEARCH METHODOLOGY

In this research, five AI algorithms which are CNN, SVM, MLP, KNN, and Random Forest were used. We managed to obtain stable output for all of these algorithms.

A. Datasets Collections

Dataset is the heart of AI algorithms. We collected and trained our model in the different datasets to check the robustness of the model. We used four datasets - Ekush, NumtaDB, BDRW, and CMATERdb.

Ekush is the largest dataset of Bangla Handwriting with 367,018 data in 122 classes. This dataset has some properties including age, gender, location, and educational status. There are almost 30,688 unique digits in the Ekush dataset, whereby 15,622 data is collected from male writers and 15,065 data is collected from the female writers. This dataset can be used on detecting age, gender and location from handwriting. Figure 2 shows the distribution of the different classes in the Ekush database.

The NumtaDB dataset is one of the most enriched datasets with more than 85,000 samples in it. The dataset was compiled by Bengali.AI and they build this dataset by merging six different data sources. Table II shows the frequency of each datasets. And Figure 3 shows the distribution of the different classes in the NumtaDB database.

CMARtdb is the oldest dataset of Bangla characters. This dataset was built by 'Center for Microprocessor Applications for Training Education and Research' (CMATER) research lab at Jadavpur University. The dataset has 6,000 labeled Bangla Handwriting samples. In this dataset, all classes are equally distributed where each class contains 600 32x32 pixel

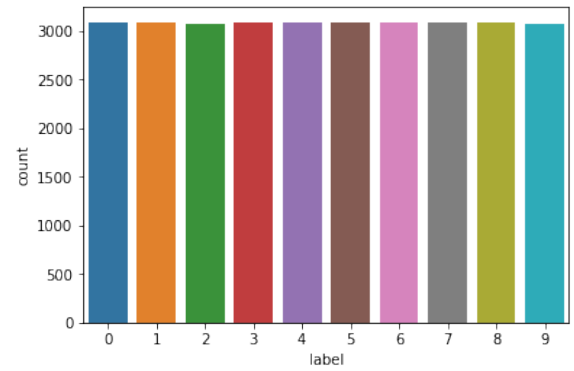


Fig. 2: Distribution of the classes in the Ekush database.

TABLE II: Data sources

Name of the Dataset	Contributing Institution	Number of Samples	Removed samples	File format
Bengali Handwritten Digits Database (BHDDb)	CSE, BUET	23,400	209	RGB
BUET101 Database (B101DB)	BUET	435	7	RGB
OngkoDB	CSE, BUET	28,900	321	Gray-scale
ISRTHDB ISRT,	DU	13,133	277	RGB
BanglaLekha-Isolated	-	20,319	572	Binary
UIUDB	UIU	576	81	RGB

3 channel BMP format images. Images are clean, without noise, without any overwrite, the edge is blocky and all images are correctly labeled. Figure 4 shows the class distribution of CMARtdb dataset.

BDRW (Bengali Digit Recognition in the Wild) [5] is an open-sourced dataset, hosted in Kaggle. The main feature of BDRW is that it is collected from real-life Bangla scripts, such as books, posters, etc. With 1,393 images, it is one of

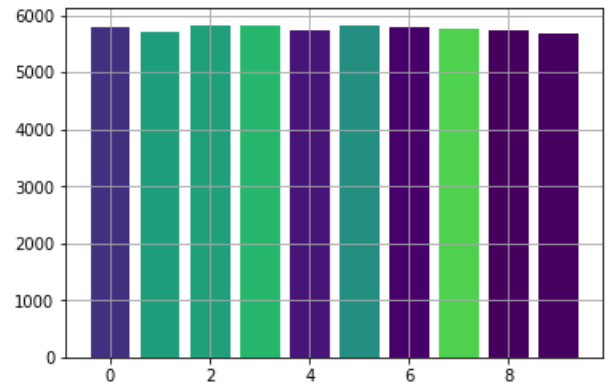


Fig. 3: Distribution of the classes in the NumtaDB database

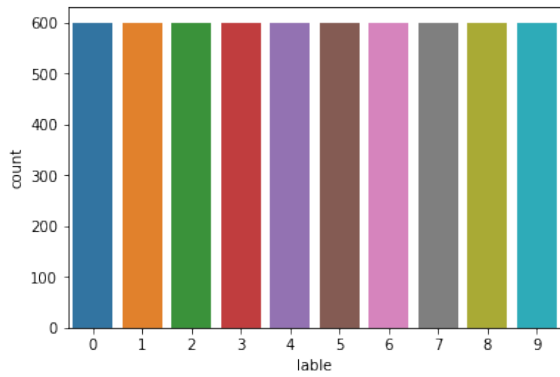


Fig. 4: Distribution of the digits in the CMARtdb database

the best datasets in terms of quality. The dataset was released on IEEE TechSym 2016. Figure 5 shows the distribution of the different classes in the BDRW database. Meanwhile, Figure 6 shows some sample images from these datasets.

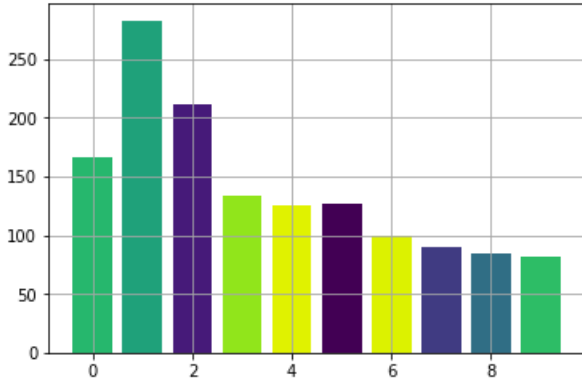


Fig. 5: The distribution of the classes in the BDRW database.

B. Data Processing

Data is the most important part of machine learning algorithms. But most of the data found online are not ready for machine learning algorithms. So, we need some manual processing to fit our data onto any machine learning model.

We processed all the datasets before fitting them onto our five algorithms. NumtaDB and BDRW data came with image formats, and the data labels came with .csv and .xlsx format. Thus, we need to map the images with .csv and .xlsx to predict the output.

In CMARtdb, the dataset images were put into a folder, and the folder name represents the class number. We read those images from each folder and give them a corresponding class based on their folder name. CMARtdb dataset contains a 32x32 pixel one channel grayscale image.

Ekush dataset came with two versions, one with images and another one is in .csv format. For the image version, each image was labeled by serial number, age, gender, location, and educational status, and each class was put into a separate

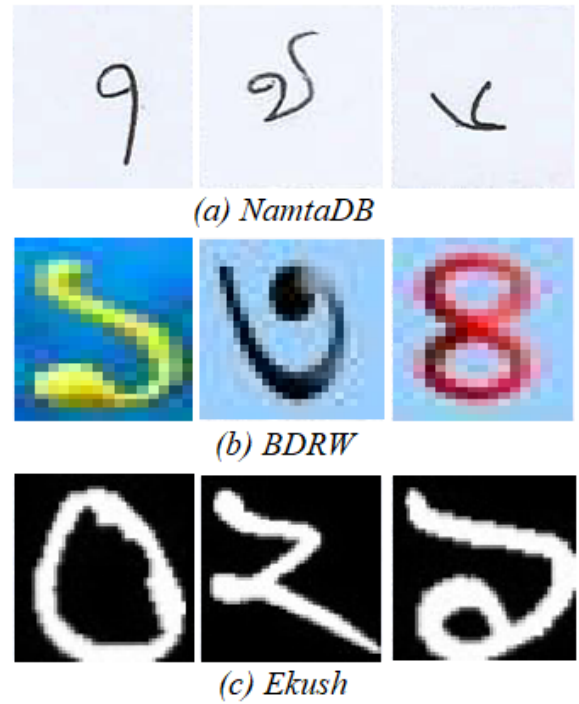


Fig. 6: Example of dataset

folder. In this research, we mainly used the .csv version of the dataset, whereby each 28 x 28-pixel data is put into 785 rows, which 784 rows are for images data and one for the label.

C. Data Augmentation

Data augmentation often helps to produce artificially more data which significantly improves the accuracy of machine learning algorithms. In handwritten digit, data augmentation helps positively, because a single person can write a digit in many different variances. We used several techniques to augment the data of our training sample, including:

- Randomly rotating the training images by 10 degrees.
- Shifting the width of the images by 10%.
- Shifting the height of the images by 10%.
- Using min-max normalization to reduce the noise and eliminate the effect of lights. Equation 1 is used for Min-Max normalization. In this normalization, all the values are mapped between 0 to 1. Normalization helps to reduce the noise and effect of lighting from the image.

$$Z_i = \frac{X_i - \text{minimum}(X)}{\text{maximum}(X) - \text{minimum}(X)} \quad (1)$$

D. Building Training & Testting Sets

For all datasets, the data was divided by 80%-20% (Train-Test Split) if they are not already separated like CMARtdb. Training data is completely independent of testing data.

IV. ALGORITHM EVALUATION AND RESULTS

After all necessary pre-processing and augmentation, we applied all data to the five machine learning algorithms, including K-Nearest Neighbor (KNN), Support Vector Machine

(SVM), Random Forest (RF), Multi-layer Perceptron (MLP) and, Convolutional Neural Network (CNN).

A. K-Nearest Neighbour

K-nearest neighbors are used for classification and regression. It is a non-parametric algorithm. We got the highest accuracy if we set the number of neighbors as 5. The performance of KNN is shown in Table III. Figure 7 is showing the confusion matrix for all datasets.

TABLE III: KNN Accuracy in different dataset

Name of the Dataset	Accuracy	Precision	Recall	F1-Score
Numta	73.73%	77.90%	73.34%	73.07%
BDRW	78.14%	84.28%	74.83%	77.25%
Ekush	89.91%	90.38%	89.95%	89.87%
CMARTdb	94.91%	94.51%	94.47%	94.40%

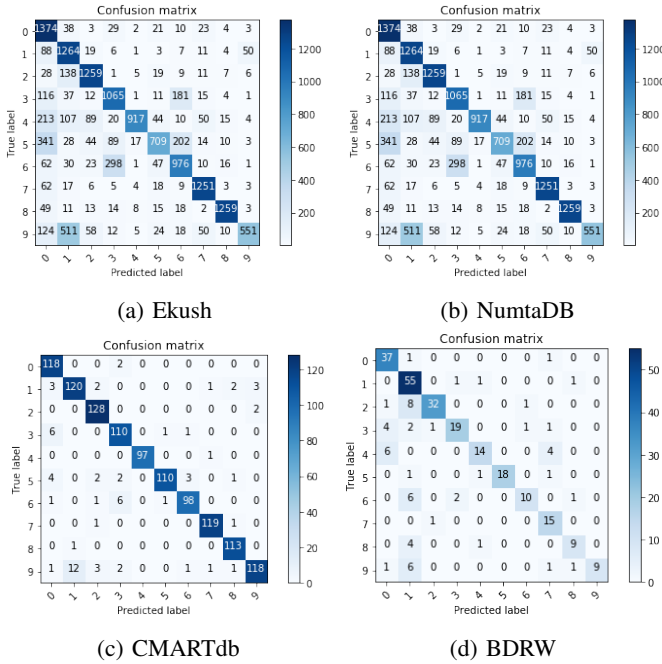


Fig. 7: Confusion Matrix for KNN

B. Support Vector Machine (SVM)

Support Vector Machine (SVM) is a discriminative classifier that works by separating hyperplanes. This algorithm can be used in both regression and classification. In this research, we tried several SVM kernels, like linear, poly, rbf [13] and sigmoid [14]. Our research found that, for handwriting recognition, Radial Basis Function kernel (RBF) outperformed all other kernels. Table IV is showing the SVM algorithm accuracy on different datasets. Figure 8 is showing all the matrix outputs.

C. Multi-layer Perceptron (MLP)

A Multi-layer Perceptron (MLP) consists of at least three feed-forward layers, output and input layer along with a hidden

TABLE IV: SVM Accuracy in different dataset

Name of the Dataset	Accuracy	Precision	Recall	F1-Score
Numta	78.00%	85.48%	85.45%	85.47%
BDRW	74.55%	87.04%	68.01%	72.78%
Ekush	90.72%	91.05%	90.71%	90.77%
CMARTdb	95.01%	95%	95.02%	95%

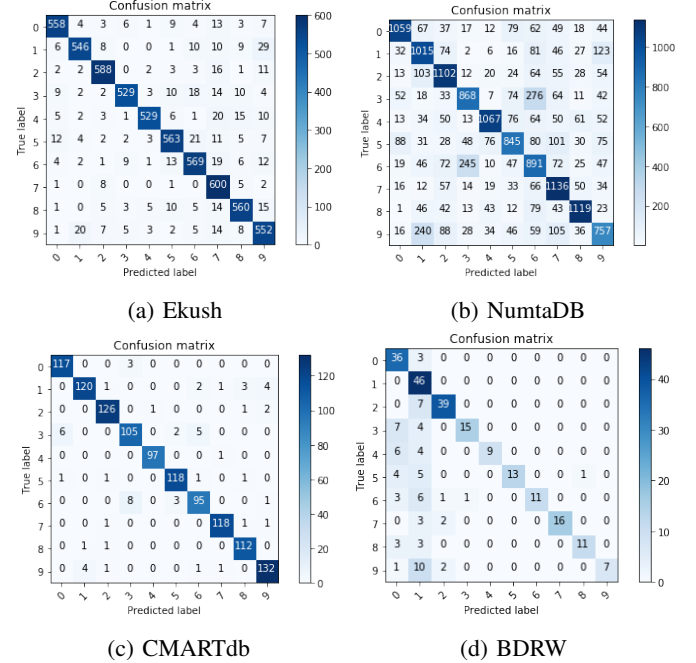


Fig. 8: Confusion Matrix for SVM

layer. Each node will have a nonlinear activation function, and back-propagation is used for training. We tried the different combinations of the model with a different combination of the optimizer, such as lbfgs [15], sgd [16], adam [17] with different learning rate methods along with different neuron size of 500 to 6,000. For 4,800 neurons with Adam optimizer, we got the best accuracy for the MLP. Table V is showing the performance of the MLP classifier. Figure 9 is showing the confusion matrix.

TABLE V: MLP Accuracy in different dataset

Name of the Dataset	Accuracy	Precision	Recall	F1-Score
BDRW	80.64%	84.31%	77.44%	79.57%
Numta	88.96%	88.47%	88.50%	88.45%
Ekush	95.02%	95.02%	95.03%	95.02%
CMARTdb	95.33%	95.40%	95.43%	95.40%

D. Random Forest (RF)

In the Random Forest algorithm, the 'forest' builds decision trees by ensemble method. This algorithm uses the bagging method for training on data. We set the number of trees in the algorithm as 100 with the criterion set as Gini. The criterion function measures the quality of a split. Supported criteria are "Gini" for the Gini impurity and "entropy" for the information

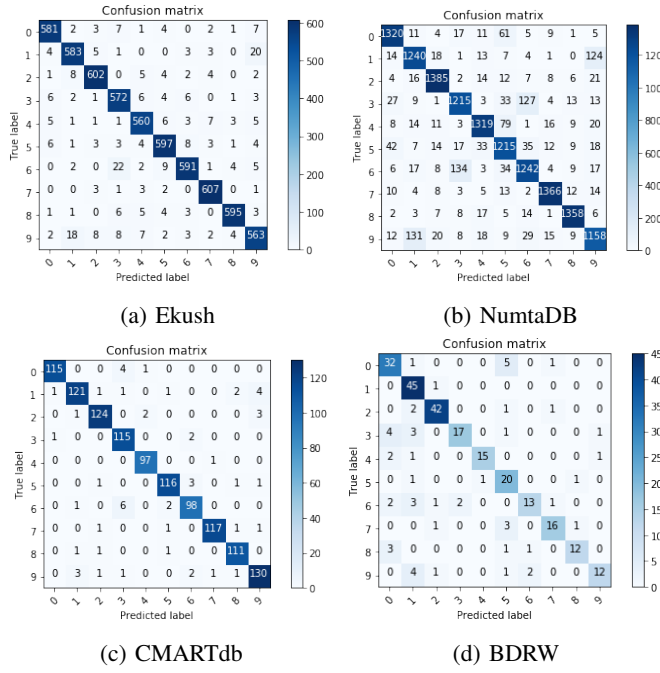


Fig. 9: Confusion Matrix for MLP

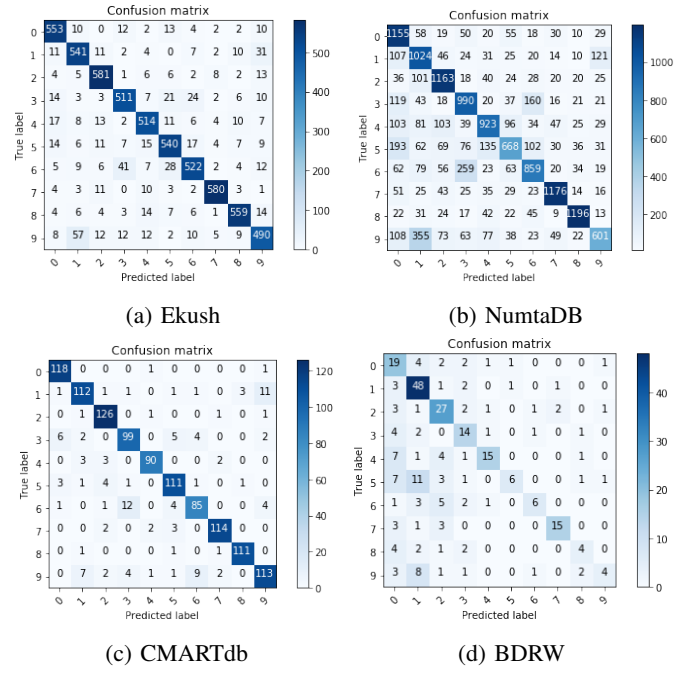


Fig. 10: Confusion Matrix for Random Forest

gain. The performance of the Random forest is shown in Table VI. Figure 10 is showing the confusion matrix.

TABLE VI: Random Forest Accuracy in different dataset

Name of the Dataset	Accuracy	Precision	Recall	F1-Score
BDRW	59.14%	60.46%	53.02%	54.85%
Numta	73.50%	68.66%	67.82%	67.57%
Ekush	87.46%	87.52%	87.51%	87.48%
CMARtdb	89.92%	89.99%	89.98%	89.94%

E. Convolutional Neural Network (CNN)

For all datasets, we build Lenet type architecture for training. The first two 5x5 convolutions were used along with two ReLU (2) activation layer and 2x2 max-pooling were applied. Then two 3x3 convolution with the ReLU layer, after that max-pooling layer was 2x2. Later again two 3x3 convolutions with the ReLU layer, after that max-pooling layer was 2x2. In order to develop the model dense layer or fully connected 64 layer neuron, the ReLU layer was applied. Finally, dropout [18] layer with 20% and the output layer has 10 output as ten classes (0-9). In the entire model, the value of stride was 1 and the padding value was 0. Finally, as we have ten-class categorical cross-entropy (3) used for loss function and ‘Adam’ (4) optimizer used for optimization. Figure 11 show’s the Sankhya CNN architecture.

$$\text{ReLU}(X) = \text{MAX}(0, X) \quad (2)$$

$$L_i = - \sum_j t_{i,j} \log(p_{i,j}) \quad (3)$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \varepsilon} \hat{m}_t \quad (4)$$

The test accuracy of the CNN model is shown in Table VII, and from our comparison, CNN outperforms all other algorithms. Figure 12 shows the confusion matrix.

TABLE VII: CNN Accuracy in different dataset

Name of the Dataset	Accuracy	Precision	Recall	F1-Score
BDRW	96.65%	94.18%	94.62%	94.19%
Numta	98.94%	98.80%	98.80%	98.80%
CMARtdb	99.25%	99.20%	99.13%	99.16%
Ekush	99.71%	99.71%	99.71%	99.71%

V. RESULT & COMPARISON

Table VIII, Table IX and Table X show an overall comparison of the five algorithms on these databases based on accuracy, precision and recall respectively. It was clear that the CNN algorithm had better performance than the other algorithms. It was also clear that BDRW is a complex database to recognize compared to the Ekush and NumtaDB. This was expected because of the low amount of data and the variance of the images.

A. Reason For CNN Outperforming then Other Algorithms

CNN algorithm could focus on the local patterns, and for that reason, CNN was able to extract important features of an image more easily than a classic multilayer perceptron. One of the core building blocks of CNN is the convolution layer which identified the low-level features. In the model, max-pooling was used to reduce the dimensionality of the feature

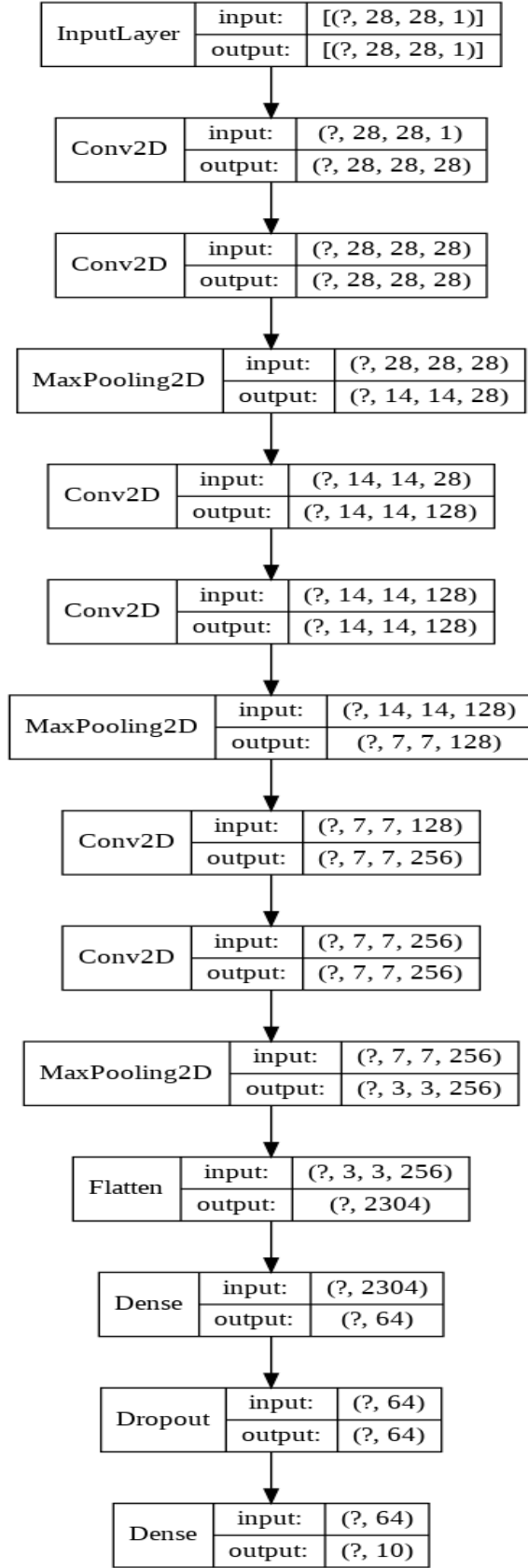


Fig. 11: Proposed CNN Model

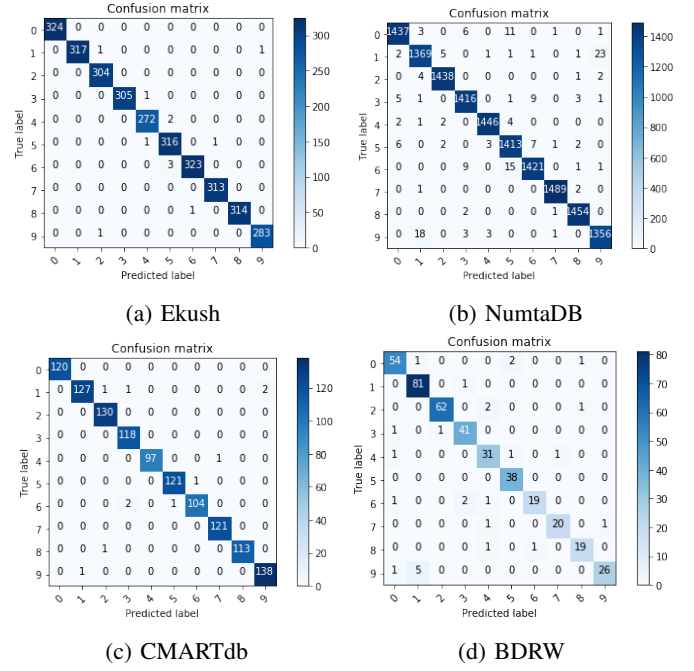


Fig. 12: Confusion Matrix for CNN

TABLE VIII: Accuracy Comparison Between Different Algorithms

Algorithm	Accuracy for Ekush	Accuracy for NumtaDB	Accuracy for BDRW	Accuracy for CMARtdb
CNN	99.71%	98.82%	96.65%	99.25%
MLP	94.89%	88.96%	80.82%	95.33%
SVM	90.72%	78.00%	74.55%	95.01%
RF	87.46%	73.50%	59.14%	89.92%
KNN	89.91%	73.13%	78.14%	94.91%

and controlling the overfitting. The dropout percentage is 20% and it plays an important role to find the best matching feature by controlling the overfitting [18]. From Figure 13's Training vs Validation curve for CNN, we can tell that our proposed CNN model didn't suffer any overfitting issues.

B. Result Comparison with previous works

We also compared our handwritten digit recognition work reported elsewhere and found our algorithms outperform in some cases. Table XI shows some of the best works on NumtaDB and CMARtdb datasets.

VI. FUTURE WORK & CONCLUSION

Random Forest and, SVM can be used with features from CNN to yield a better prediction accuracy compared to the original CNN. After extracting features from both of the NumtaDB and BDRW datasets, it should be possible to use Random Forest, SVM. The next approach can be implemented with other classifiers on the extracted features as this approach can perform better than the original CNN.

Sankhya has benchmarked five well-studied AI algorithms on all the publicly available datasets along with the recently

TABLE IX: Precision Comparison Between Different Algorithms

Algorithm	Precision for Ekush	Precision for NumtaDB	Precision for BDRW	Precision for CMARtdb
CNN	99.71%	98.80%	94.18%	99.20%
MLP	95.02%	88.47%	84.31%	95.40%
SVM	91.05%	85.48%	87.04%	95.00%
RF	87.52%	68.66%	60.46%	89.99%
KNN	90.38%	77.90%	84.28%	94.51%

TABLE X: Recall Comparison Between Different Algorithms

Algorithm	Recall for Ekush	Recall for NumtaDB	Recall for BDRW	Recall for CMARtdb
CNN	99.71%	98.80%	94.62%	99.13%
MLP	95.03%	88.50%	77.44%	95.43%
SVM	90.71%	85.45%	68.01%	95.02%
RF	87.51%	67.82%	53.02%	89.98%
KNN	89.95%	73.34%	74.83%	94.47%

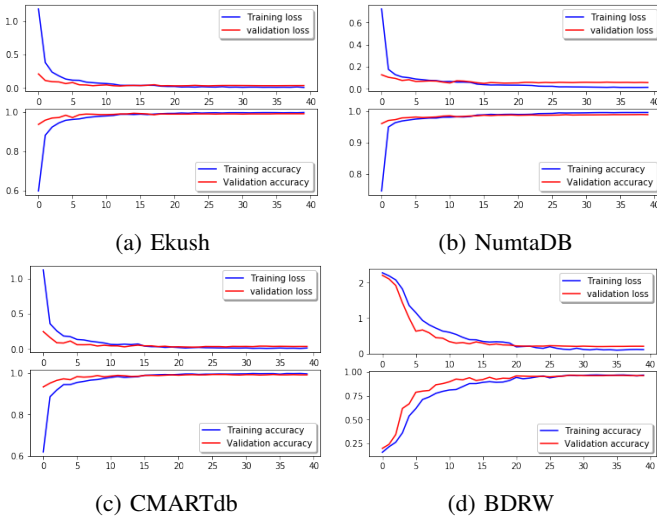


Fig. 13: Training vs Validation curve for CNN

published Ekush and NumtaDB databases. The goal of the study is to build a first level benchmark, on top of which more aggressive tuning can be done to get better results and to build better models.

ACKNOWLEDGEMENT

We thank our colleague Nor Azriah Aziz, who provided insight and expertise in proofreading and formatting that greatly assisted this research.

REFERENCES

- [1] R. Banerji, "Origin of the bengali script," Aug 1919. [Online]. Available: <https://www.universallibrary.org/details/in.ernet.dli.2015.277219>
- [2] A. S. A. Rabby, S. Haque, M. S. Islam, S. Abujar, and S. A. Hossain, "Ekush: A multipurpose and multitype comprehensive database for online off-line bangla handwritten characters," in *Recent Trends in Image Processing and Pattern Recognition*, K. C. Santosh and R. S. Hegadi, Eds. Singapore: Springer Singapore, 2019, pp. 149–158.
- [3] S. Alam, T. Reasat, R. M. Doha, and A. I. Humayun, "Numtdb - assembled bengali handwritten digits," *CoRR*, vol. abs/1806.02452, 2018. [Online]. Available: <http://arxiv.org/abs/1806.02452>

TABLE XI: Comparison Between Different Works

NumtaDB Work	Accuracy	CMATERdb Work	Accuracy
Shawon et al. [19]	92.72%	Hassan et al. [20]	96.7%
Hasib et al. [21]	97.09%	Sarkhel et al. [22]	98.23%
Mamunur et al. [23]	96.69%	Hybrid-HOG [24]	99.17%
Mahmudul et al. [25]	99.34%	Rabby et al. [7]	99.42%
Proposed Sankhya model	98.82%	Proposed Sankhya model	99.25%

- [4] R. Sarkar, N. Das, S. Basu, M. Kundu, M. Nasipuri, and D. K. Basu, "Cmaterdb1: a database of unconstrained handwritten bangla and bangla-english mixed script document image," *International Journal on Document Analysis and Recognition (IJ DAR)*, vol. 15, no. 1, pp. 71–83, Mar 2012.
- [5] D. Sheet, "Bengali digit recognition in the wild (bdrw)," Aug 2016. [Online]. Available: <https://www.kaggle.com/debdoot/bdrw>
- [6] U. Bhattacharya and B. B. Chaudhuri, "Handwritten numeral databases of indian scripts and multistage recognition of mixed numerals," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 3, pp. 444–457, March 2009.
- [7] A. S. A. Rabby, S. Abujar, S. Haque, and S. A. Hossain, "Bangla handwritten digit recognition using convolutional neural network," in *Emerging Technologies in Data Mining and Information Security*, A. Abraham, P. Dutta, J. K. Mandal, A. Bhattacharya, and S. Dutta, Eds. Singapore: Springer Singapore, 2019, pp. 111–122.
- [8] M. Biswas, R. Islam, G. K. Shom, M. Shopon, N. Mohammed, S. Momen, and A. Abedin, "Banglalekha-isolated: A multi-purpose comprehensive dataset of handwritten bangla isolated characters," *Data in Brief*, vol. 12, pp. 103 – 107, 2017.
- [9] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov 1998.
- [10] K. C. Santosh and L. Wendling, "Character recognition based on non-linear multi-projection profiles measure," *Frontiers of Computer Science*, vol. 9, no. 5, pp. 678–690, Oct 2015.
- [11] S. Haque, A. S. A. Rabby, M. S. Islam, and S. A. Hossain, "Shonkhanet: A dynamic routing for bangla handwritten digit recognition using capsule network," in *Recent Trends in Image Processing and Pattern Recognition*, K. C. Santosh and R. S. Hegadi, Eds. Singapore: Springer Singapore, 2019, pp. 159–170.
- [12] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," *CoRR*, vol. abs/1710.09829, 2017. [Online]. Available: <http://arxiv.org/abs/1710.09829>
- [13] J.-P. Vert, K. Tsuda, and B. Schölkopf, "1 a primer on kernel methods," 2004.
- [14] H.-T. Lin and C.-J. Lin, "A study on sigmoid kernels for svm and the training of non-psd kernels by smo-type methods," *Neural Computation*, 06 2003.
- [15] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1, pp. 503–528, Aug 1989. [Online]. Available: <https://doi.org/10.1007/BF01589116>
- [16] J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, Eds., *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*. Curran Associates, Inc., 2008. [Online]. Available: <http://papers.nips.cc/book/advances-in-neural-information-processing-systems-20-2007>
- [17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: <http://dblp.uni-trier.de/db/journals/corr/corr1412.html#KingmaB14>
- [18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014. [Online]. Available: <http://jmlr.org/papers/v15/srivastava14a.html>

- [19] A. Shawon, M. Jamil-Ur Rahman, F. Mahmud, and M. M. Arefin Zaman, "Bangla handwritten digit recognition using deep cnn for large and unbiased dataset," in *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, Sep. 2018, pp. 1–6.
- [20] T. Hassan and H. A. Khan, "Handwritten bangla numeral recognition using local binary pattern," in *2015 International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)*, May 2015, pp. 1–4.
- [21] H. Zunair, N. Mohammed, and S. Momen, "Unconventional wisdom: A new transfer learning approach applied to bengali numeral classification," in *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, Sep. 2018, pp. 1–6.
- [22] R. Sarkhel, N. Das, A. K. Saha, and M. Nasipuri, "A multi-objective approach towards cost effective isolated handwritten bangla character and digit recognition," *Pattern Recogn.*, vol. 58, no. C, pp. 172–189, Oct. 2016. [Online]. Available: <https://doi.org/10.1016/j.patcog.2016.04.010>
- [23] M. Rahaman Mamun, Z. Al Nazi, and M. Salah Uddin Yusuf, "Bangla handwritten digit recognition approach with an ensemble of deep residual networks," in *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, Sep. 2018, pp. 1–4.
- [24] S. M. A. Sharif, N. Mohammed, N. Mansoor, and S. Momen, "A hybrid deep model with hog features for bangla handwritten numeral classification," in *2016 9th International Conference on Electrical and Computer Engineering (ICECE)*, Dec 2016, pp. 463–466.
- [25] M. Mahmudul Hasan, M. Rafid UI Islam, and M. Tareq Mahmood, "Recognition of bengali handwritten digits using convolutional neural network architectures," in *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, Sep. 2018, pp. 1–6.