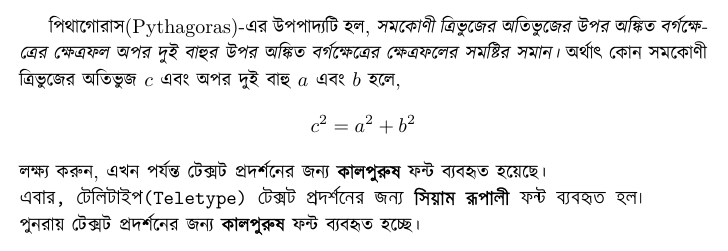
A Novel Pair-wise Language Detection Approach using Convolutional Neural Network Specifically Targetting Bangla and English

*Abstract*—Language detection is an essential pre-processing step in the implementations of many multilingual document-processing solutions, such as Optical Character Recognition (OCR) and machine translation. Specifically, language detection research for Bangla is very rare, with only a handful of solutions ever reported in the literature. In this paper, we present a novel, lightweight, small footprint convolutional neural network, which detects Bangla and English languages—directly from scanned mixed-language document images. The proposed model achieves 99.98% recognition accuracy for this specific two-language classification problem.

Keywords—Language Detection, Bangla, English, CNN, Pattern Recognition.

# Introduction

Detecting the language of a mixed-language document is a very important pre-processing step in automated document processing tasks. One such processing task is Optical Character Recognition (OCR). Another classic example is machine translation where detecting the language is an essential pre-requisite. Humans are great at detecting languages, but for a machine, the solution is non-trivial.

There is another large difference between the two tasks mentioned above. In machine translation tasks, the texts are, in almost all the cases, already in electronic format. An example is the widely used Google Translate page (https://translate.google.com.bd), where the language is detected automatically. In OCR tasks, the source is in image format, and no such electronic transcription is available, making the task harder. The problem here is that the cause-and-effect is circular—we can detect the language better if we can run an OCR first, but to use the correct OCR, we need to know what the language is first. Although this paper primarily focuses on the latter problem—applying language detection specifically within the OCR domain, it can also be used within a machine translation domain, albeit with some minor modifications.

Nowadays, OCR systems have been developed for most languages. For OCR, the first step is to perform layout analysis, followed by segmentation (paragraph segmentation, word segmentation, character segmentation, etc.), for which language detection is an essential pre-requisite. Specifically, within the domain of Bangla language documents, the mixture of Bangla and English is not uncommon at all as seen in Fig 1, making the task of Bangla document processing comparatively harder. Bangla is one of the most widely used languages in the world, spoken as the first language by 160 million people. For another 20 million speakers, it is used as a second language [1]. With over 2 billion people speaking and using English in their daily lives [2], matured English OCR solutions are widely available—both for academic research and commercial applications. Sadly that is not true for Bangla. There are commercial solutions such as Google Lens and Google Tesseract [3] that work on multiple languages, including English and Bangla. Still, the accuracy of these tools is not good enough to be used in serious Bangla OCR applications. To develop a commercially-viable Bangla OCR, we must address the elephant in the room and solve the language detection problem first.

In this paper, we proposed a novel Convolutional Neural Network (CNN) model to identify Bangla and English languages from mixed-language scanned documents.

Fig. 1: Bangla and English mixed in a primarily Bangla document

# Review

There is not much reported work for language detection in scanned document images. We found a couple of works, including a paper from Dr. N. Jayanthi et al. [4]. They proposed a convolutional neural network solution for language detection on Hindi, Tamil, and English handwritten characters. Their training dataset consisted of 39,000 handwritten images of characters from three languages, written in varying font-weight, angles, and coverage. Eighty percent of characters were used for training, and the other 20% was used for testing the model accuracy of character classification. They achieved training accuracy of 78% in 40 learning epochs and 74% for the testing accuracy.

In Oaknorth [5], the authors use a CNN model to detect language and focus on 3 languages—German, English, and Italian. They collected book samples and cropped them randomly into 150 x 150 pixels resulting in a corpus of 2,090 images for English, 1,910 images for German, and 1,680 images for Italian; the total number of samples was 5,680. They split the corpus into two parts—80% for training and 20% for testing. They reported running their model for 100 epochs, with early stopping and report a 97% accuracy on the test set within 34 epochs.

Another widely used solution is the language detection library within Google Tesseract. Tesseract is used at page-level documents, after converting into text and then using a library; it predicts languages and returns the most used language in the document.

Although, we did not find that much work with scanned documents, but we did find additional research with speech and audio to detect languages.

Revay et al. [6] used audio spectrograms images to train a CNN. Their audio file duration was 3.75 seconds, and they used six different languages: English, Spanish, French, German, Russian, and Italian. They used 5,000 clips per language for the training set and 2,000 clips for the validation set; in total, 60,000 samples were used in this reported research. They report an accuracy of 97% on binary language classification. For multiclass classification with six languages, the report an accuracy of 89%.

C. Bartz et al. [7] captured spatial information from audio snippets using a CNN through a sequence of time steps used by a Recurrent Neural Network (RNN). They use two data sets, one is the European Speech dataset [8], and the other is the YouTube News dataset, such as official BBC News [9]. The split their dataset into three parts: a training set, a validation set, and a testing set in the ratio of 70%, 20%, and 10% respectively. In the European Speech dataset, 19,000 training samples are available of 53 hours of speech audio, and the YouTube News dataset has 194,000 training sample of 540 hours of speech audio. For the YouTube dataset, they achieved a 90% accuracy for the proposed vanilla CNN model and 98% for the Convolutional Recurrent Neural Network (CRNN) model, and in the European Speech dataset, their vanilla CNN model and CRNN model achieved accuracies of 90% and 91% respectively.

# Dataset

To train our model, we built our own dataset. We used two publicly available datasets for Bangla and English handwritten words. CMATERdb [10] word dataset is used for handwritten Bangla word, and IAMonDo-database [11] is used for English handwriting words. For printed words, we collected our own dataset. We collected English pages from an open-source Kaggle repository [12] and created a Bangla word document with different fonts and sizes. Details are presented in the following sections.

## Dataset Preprocessing

IAMonDo database has 115,320 word images, and the CMATERdb dataset has 17,079 word images.

English printed dataset has 14,025 word images.

To build our printed Bangla corpus, we decided to use almost the same number of images as above, which is 15,000. Our dataset has 15,001 and 15,002 images of Bangla, and the English handwritten words, respectively. We also have 13,394 and 14,025 scanned word images, respectively, for Bangla and English printed documents. We merged both handwritten and printed datasets and built one combined dataset, which has 28,395 Bangla words and 29,027 English words. Table 1 shows a summary of the dataset.

1. Summary of the Datasets

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Language | Type | Original Size | Processed Size | Merged Size |
| Bangla | Printed | 13,394 | 13,394 | 28,395 |
| Handwritten | 17,079 | 15,001 |
| English | Printed | 14,025 | 15,002 | 29,027 |
| Handwritten | 115,320 | 14,025 |

## Train-Test-Validation Split

We split our dataset into three parts after random shuffling. We separated 13% of data for validation set, and the rest 83% was assigned for training and testing the model. Later, we separated the dataset again, where we put 23% on the test and rest for the training set. In summary, we used 13% for cross-validation, 20% for the testing, and 73% for the training set from the whole dataset. Fig 2 shows the frequency distributions for each set.

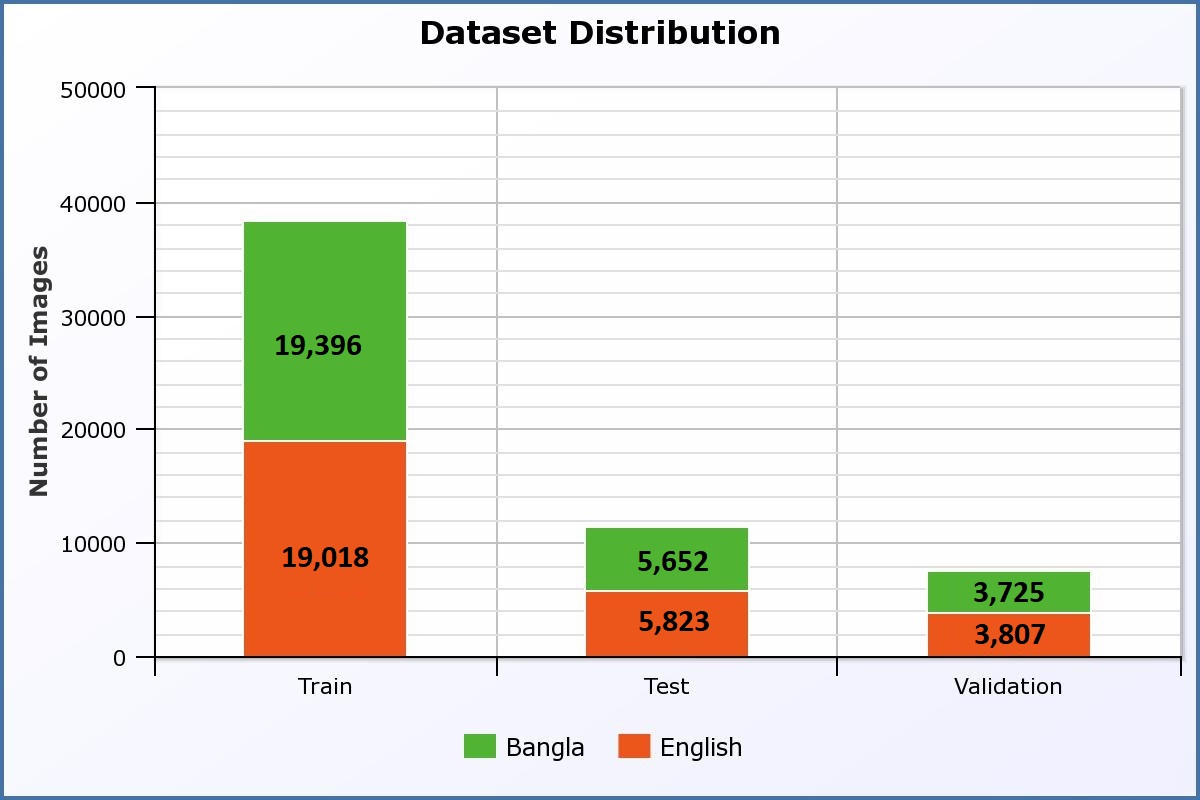


Fig. 2 Dataset distribution

## Data Augmentation

Having a large dataset is crucial for the performance of a deep learning model [13]. Through a few workarounds, we can improve the performance of the model by augmenting the data we already have. We implemented some data augmentation techniques in our dataset to increase the size, which includes:

* Rotation image 5 degrees,
* Normalize and rescaling the image using min-max normalizer,
* Shear image 10%,
* Zoom image 20%,
* Shifting the weight 10%,
* Shifting the height of 10%

Fig 3 and Fig 4 show some examples of applied data augmentation for Bangla and English datasets, respectively.

# Work Methodology

As stated already, language detection is a crucial part of developing an OCR system. A multilingual OCR system needs to identify the language so that it can apply the language-specific model for character recognition.

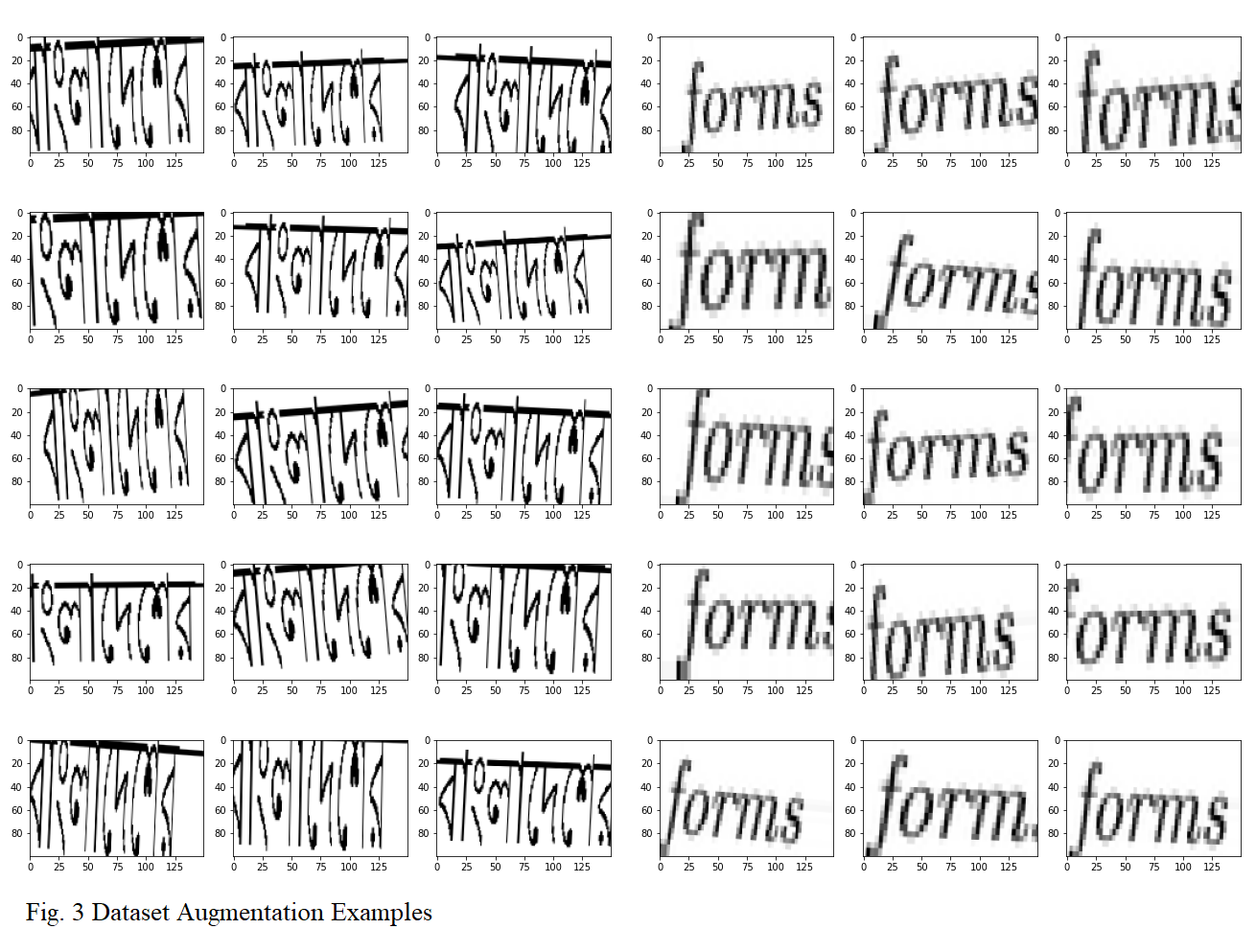


Fig. 3 Augmented samples for Bangla words

## Proposed Model

Convolution, the basic building block of Convolutional Neural Network (CNN) [14], is a mathematical combination of two functions that merges two sets of information to produce a third function. The convolution is performed on the input data using a filter (kernel) to produce a feature map. CNN can minimize the number of parameters to solve complex image recognition tasks.

The proposed model has a CNN for the classification of language identification, which has a two-class output: English and Bangla. This model used convolution, max pooling layer, fully connected dense layer, and regularization methods such as batch normalization and dropout, as seen in Fig 5.

In the model architecture, we have one convolutional layer in the first block, which is also an input layer with kernel size 3, and the filter size is 64. In this convolutional layer, the input image width is 100, and the height is 150 using the activation function ReLU [15]. The second block is a Batch Normalization [16] layer, with momentum set to default. It is connected with a max-pooling layer with a pool size of 2, followed by a 25% dropout layer. Then the output is flattened to an array and passed through a fully connected dense layer of 256 hidden units, with ReLU activation and regularized with another batch normalization layer followed by 50% dropouts layer and passed through a fully connected dense layer of 2 nodes with ‘SoftMax’ [17] activation. This final layer is out of the output layer.

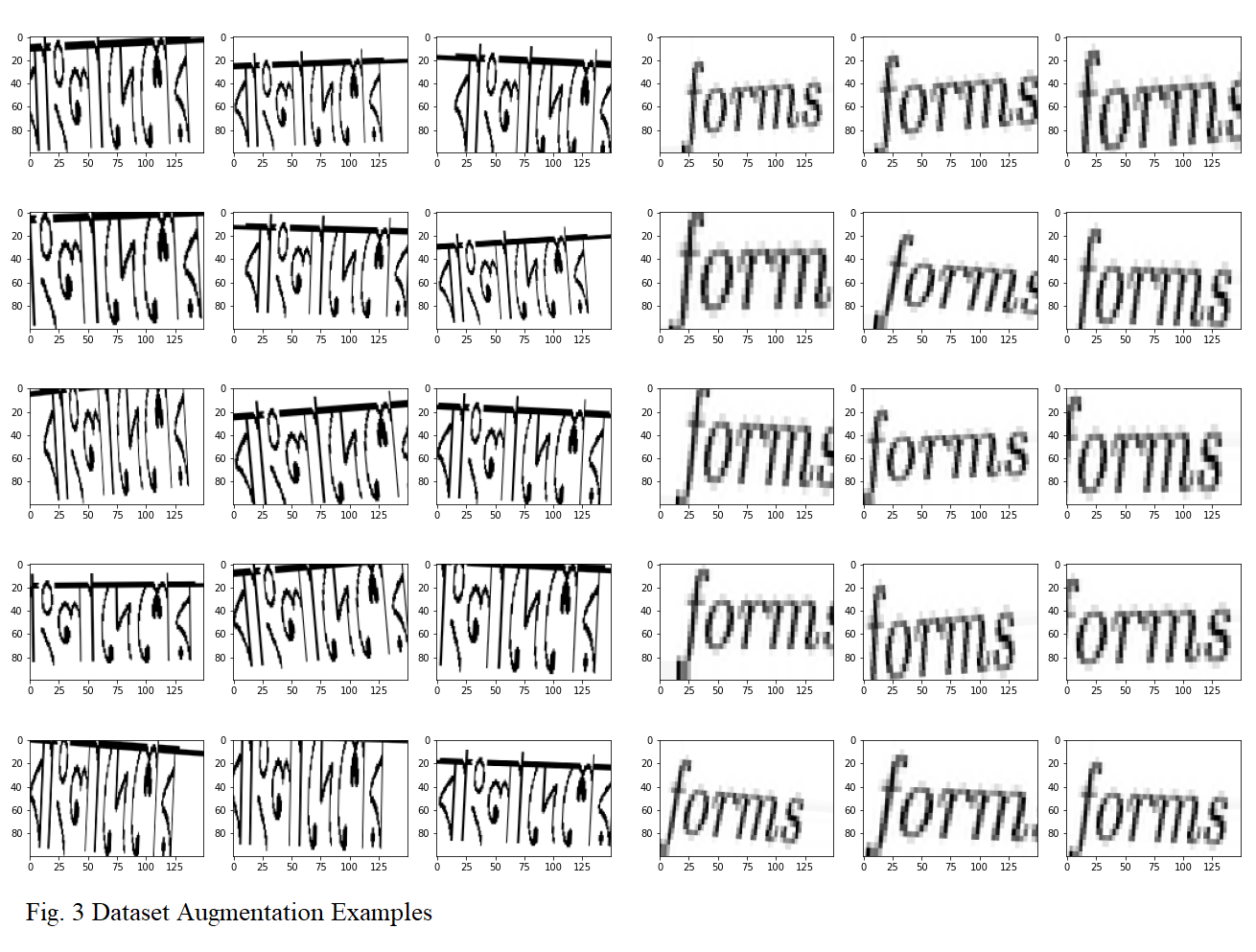


Fig. 4 Augmented samples for English words

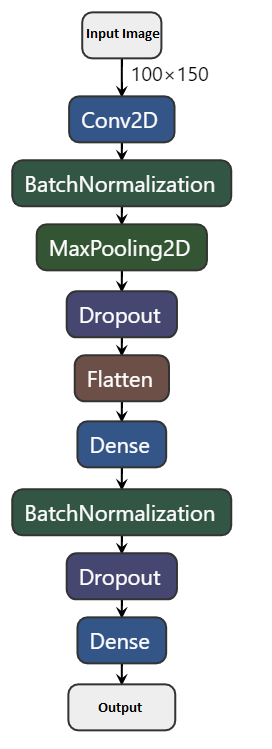


Fig. 5 Proposed CNN model

To minimize the error of the convolutional algorithms, optimization algorithms are heavily utilized. Our proposed model used Adam [18] optimizer with a learning rate of 0.001. Adam optimization algorithm is often used to update network weights iteratively in training data, which is an extension to a stochastic gradient descent algorithm. To calculate the error for optimizing algorithms, we used categorical cross-entropy function as it performs better than others [19][20].

## Epochs and EarlyStop

An issue with training neural networks is choosing how many training epochs to use. Too many epochs can result in the training dataset being overfitted, while too few can result in the model being underfitted. The EarlyStop feature has different metrics or arguments; one can adjust to set up when the training process stops.

We set our initial epochs to 25, while EarlyStop function monitored our test loss and stopped the training when needed. With this tool, our model stopped at 17 epochs, based on the model loss.

# Performace

Our proposed method achieves very good results for the train, test, and validation sets.

## Learning Curve

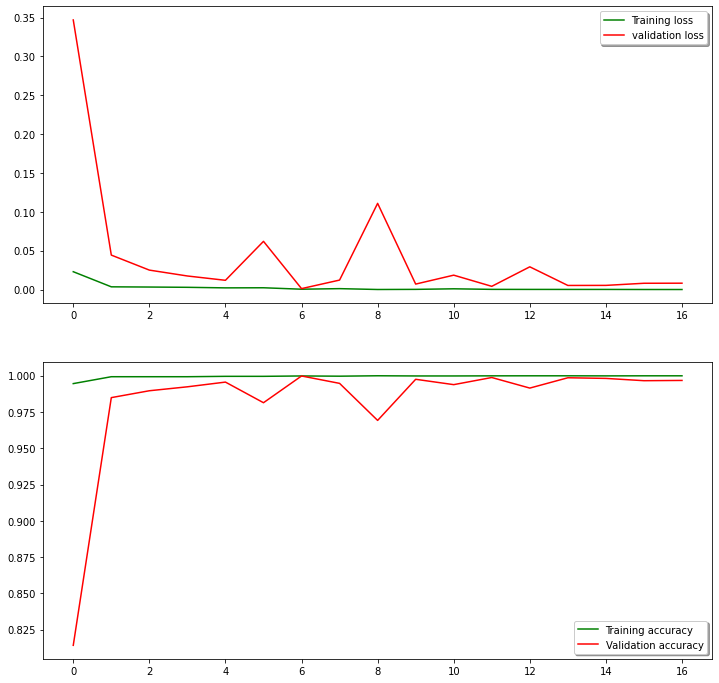
A learning curve is a plot of output model learning over time and experience. Learning curves are a commonly used machine learning testing method for algorithms, which incrementally learn from a training dataset. After each update during training, plots of the calculated results can be generated to display learning curves. The model can be tested on the training dataset and a holdout validation data set. Fig 6 shows our learning curve for train test accuracy and loss.

Fig. 6 Learning curve

On this learning curve, we can see that our model has no underfitting or overfitting issue. Over time, the model stabilized while becoming more accurate.

## Accuracy and Loss

After 17 epochs, our model reached a maximum test accuracy of 99.98%, with a minimum test loss of 0.0054. For the training set, the maximum accuracy is 99.99%, with minimum loss of 1.2912e-04.

## Confusion Matrix

A confusion matrix is a table often used to define a classification model (or “classifier”) output on a collection of test data for which the actual values are known. We observed the confusion matrices for all the datasets. The confusion matrices for the train, test, and validation set are presented in Tables II, III, and IV.

1. Confusion Matrix for Train Set

|  |  |  |  |
| --- | --- | --- | --- |
| n = 38,414 | Predicted Bangla | Predicted Bangla |  |
| Actual Bangla | 19,018 | 0 | 19,018 |
| Actual English | 93 | 19,303 | 19,396 |
|  | 19,111 | 19,303 |  |

1. Confusion Matrix for Test Set

|  |  |  |  |
| --- | --- | --- | --- |
| n = 11,475 | Predicted Bangla | Predicted Bangla |  |
| Actual Bangla | 5,652 | 0 | 5,652 |
| Actual English | 37 | 5,786 | 5,823 |
|  | 5,689 | 5,786 |  |

1. Confusion Matrix for Validation Set

|  |  |  |  |
| --- | --- | --- | --- |
| n = 7,532 | Predicted Bangla | Predicted Bangla |  |
| Actual Bangla | 3,725 | 0 | 3,725 |
| Actual English | 19 | 3,788 | 3,807 |
|  | 3,744 | 3,788 |  |

# Conclusion and Future Work

In this paper, we presented an effective model to identify Bangla and English in a mixed-language document. At the word level, we achieved a 99.98% test accuracy.

The work has primarily focused on Bangla and English language detection within the context of an OCR processing pipeline. But as mentioned before, the solution can also be used in other NLP and / or electronic or image processing tasks such as machine language, automated billboard reading, number plate detection and other problems. Also, it should be pointed out that although we chose to focus on Bangla/English pair, the solution should be extendable to any pair of languages with sufficient labelled data to retrain our models. In that sense, this is a solution for any pair-wise language detection.

Another important point to mention is the lack of comparison of our work with previous research. We were unable to get the datasets that the authors of some previous research mentioned before in this paper, which meant we were unable to compare our solution directly to theirs. In addition, we were also not able to reproduce some of these aforementioned researches, as there is not enough information to implement these algorithms and test on our corpus. Specifically, language detection focusing on Bangla/English pair, where we concentrated our work, has not been published elsewhere, which makes our work entirely novel.

In summary, we have used a large word-level dataset to train our model with excellent performance. Since our dataset is now only at the word level, our clear next step is to move the model to the page level and the character level. In addition, in the proposed model, we can only detect two languages, but in the future, we plan to adapt this to multiple-language detection.

##### Acknowledgment

The authors would like to acknowledge the encouragement and funding from the “Enhancement of Bangla Language in ICT through Research & Development (EBLICT)” project, under the Ministry of ICT, the Government of Bangladesh.

##### References

1. MustGo.com. 2020. Bengali Language - Dialects & Structure - Mustgo. [online] Available at:

<https://www.mustgo.com/worldlanguages/bengali/> [Accessed 29 May 2020].

1. En.wikipedia.org. 2020. English-Speaking World. [online] Available at: <https://en.wikipedia.org/wiki/English-speaking\_world> [Accessed 29 May 2020].
2. opensource.google. 2020. Projects – Opensource.Google. [online] Available at: <https://opensource.google/projects/tesseract> [Accessed 29 May 2020].
3. Jayanthi, N., Harsha, H., Jain, N., & Dhingra, I. S. (2020, February). Language Detection of Text Document Image. In 2020 7th International Conference on Signal Processing and Integrated Networks (SPIN) (pp. 647-653). IEEE.
4. OakNorth. 2020. Oaknorth Bank | Page Not Found - Oaknorth. [online] Available at: <https://www.oaknorth.com/cnn-blog/> [Accessed 29 May 2020].
5. Revay, S., & Teschke, M. (2019). Multiclass language identification using deep learning on spectral images of audio signals. arXiv preprint arXiv:1905.04348.
6. Bartz, C., Herold, T., Yang, H., & Meinel, C. (2017, November). Language identification using deep convolutional recurrent neural networks. In International Conference on Neural Information Processing (pp. 880-889). Springer, Cham.
7. Speech Repository - European Commission. 2020. Speech Repository. [online] Available at: <https://webgate.ec.europa.eu/sr/> [Accessed 28 May 2020].
8. YouTube. 2020. BBC News. [online] Available at: <https://www.youtube.com/user/bbcnews> [Accessed 28 May 2020].
9. Indermühle, E., Liwicki, M., & Bunke, H. (2010, June). IAMonDo-database: an online handwritten document database with non-uniform contents. In Proceedings of the 9th IAPR International Workshop on Document Analysis Systems (pp. 97-104)
10. Sarkar, R., Das, N., Basu, S., Kundu, M., Nasipuri, M., & Basu, D. K. (2012). CMATERdb1: a database of unconstrained handwritten Bangla and Bangla–English mixed script document image. International Journal on Document Analysis and Recognition (IJDAR), 15(1), 71-83.
11. Kaggle.com. 2020. *Denoising Dirty Documents | Kaggle*. [online] Available at: <https://www.kaggle.com/c/denoising-dirty-documents/data> [Accessed 29 May 2020].
12. Dvornik, N., Mairal, J., & Schmid, C. (2019). On the importance of visual context for data augmentation in scene understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*
13. Albawi, Saad & Abed MOHAMMED, Tareq & ALZAWI, Saad. (2017). Understanding of a Convolutional Neural Network.

10.1109/ICEngTechnol.2017.8308186.

1. Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)* (pp. 807-814).
2. Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.
3. Goodfellow, I., Bengio, Y., and Courville, A. (2016).Deep Learning. Page 184. MIT Press.http://www.deeplearningbook.org.
4. Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
5. Golik, P., Doetsch, P., & Ney, H. (2013, August). Cross-entropy vs. squared error training: a theoretical and experimental comparison. In Interspeech (Vol. 13, pp. 1756-1760).
6. Mannor, S., Peleg, D., & Rubinstein, R. (2005, August). The cross-entropy method for classification. In Proceedings of the 22nd international conference on Machine learning (pp. 561-568).