A Deep Learning Solution to Detect Text-types using a Convolutional Neural Network

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

**Keywords:** Text-type Detection, Handwritten Content, Printed Content, CNN, Pattern Recognition.

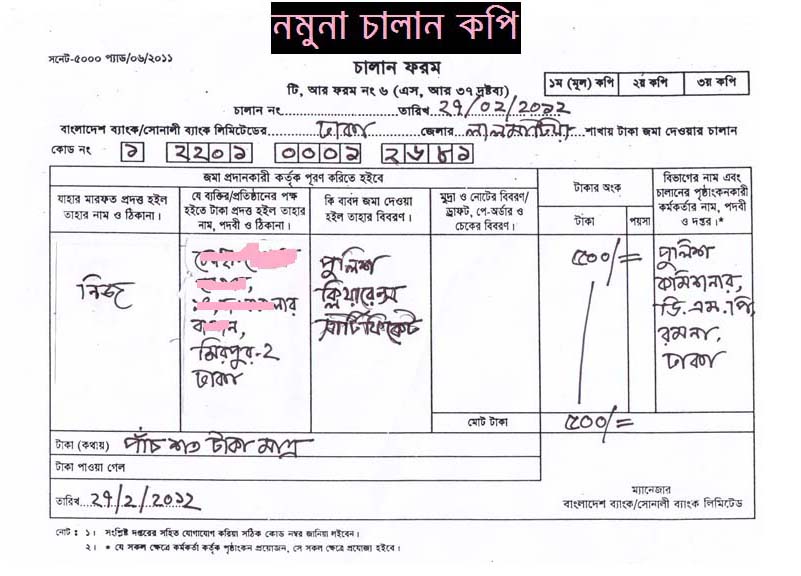
1. Introduction

OCR systems are a pipeline process, many processes aggregated to achieve the final output. These can be mainly separated into three stages: pre-processing stage (e.g. detection of languages), processing stage (e.g. segmentation), and post-processes stage (e.g. correction based on language model). As part of the pre-processing stage, one task is very important: detecting the mixture of handwritten verses printed content. Humans are great at detecting text-type, but for a machine, the solution is non-trivial.

Bangla is the national language of Bangladesh and second most widely spoken language of India, with 228 million native speakers and another 37 million as second language speakers [1][2], making it the fifth most-spoken native language and the seventh most spoken language by a total number of speakers in the world [3]. However, Bangla is considered to be a resource-poor language, as even the most basic resources of language automation such as OCRs, language models, parts of speech tagger, etc. are scarcely available.

Since the adoption of electronic workflow in offices and courts and many other segments of the country is still largely manual, many documents end up having handwritten components on top of the printed components. The ability to detect this type of variation is also very important for the design of any commercially successful Bangla OCR. A typical example is shown in Fig 1.

In this paper, our assumption is that we are part of an OCR processing pipeline and we have a functional word label segmentation module. So, we designed a Convolutional Neural Network (CNN) model to identify printed and handwritten content at the word-level from mixed-type scanned documents. We structured the rest of the paper on this assumption.



**Fig. 1.** Handwritten and printed content mixed in a primarily Bangla document

1. Literature Review

There is not much reported work for content type detection in scanned document images. We found a couple of works, including a paper from K. Zagoris et al. [4] proposed a “Bag of Visual Words” (BoVW) model, which helps to identify and separate handwritten text from machine printed text. Firstly, they detected written text followed by descriptor calculations for each block based on BoVW. Finally, they identified the blocks as either handwritten, machine printed, or noise using a binary Support Vector Machine classifier. They used PRImA-NHM1, PRImA-UIBK1, and IAM handwriting datasets and achieved an overall performance of 84%, 92.2%, and 98.9% respectively.

Kavallieratou et al. [5] proposed a discriminant analysis method to separate machine printed and handwritten text, specifically focusing on IAM-DB and GRUHD databases, for English and Greek respectively. In their process pipeline, they preprocess the document images for page skew angle correction, then text localization, then area skew angle correction, and finally, line segmentation. After these preprocessing steps, they run their proposed method and classify printed and handwritten text. They report that their proposed method achieved an average classification accuracy of 98.2%.

A Microsoft team has proposed a transfer learning method to identify handwritten documents [6]. They published their project with Jupiter notebook using Azure Machine Learning (AML) package for Computer Vision (AMLPCV) model trained on the Coco Common Object in Context dataset containing more than 200,000 labeled images and 1.5 million object instances across 80 categories. They used Visual Object Tagging Tool (VOTT) to manually label a small set of public government contract data containing both machines printed text and handwriting and labeled two classes of handwriting objects in the VOTT tool – signatures and non-signature. Using this set of labeled data, they trained a custom handwriting detection model, which can detect handwritten words from a document image with promising accuracy.

1. Datasets

We used two publicly available datasets for Bangla and English handwritten words. CMATERdb [7] word dataset is used for handwritten Bangla words, and IAMonDo-database [8] is used for English handwriting words. For printed words, we collected our own data. We collected English pages from an open-source Kaggle repository [9] and created a Bangla word document with different fonts and sizes. Details are presented in the following sections.

* 1. 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 15,001 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 Bangla and English datasets and built one combined dataset, which has 27,419 printed words and 30,002 handwritten words. Table 1 shows a summary of the dataset.

**Table 1.** Summary of the Datasets

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Language | Type | Original Size | Processed Size | Merged Size |
| Printed | Bangla | 13,394 | 13,394 | 27,419 |
| English | 14,025 | 14,025 |
| Handwritten | Bangla | 17,079 | 15,001 | 30,002 |
| English | 115,320 | 15,001 |

* 1. Train-Test-Validation Split

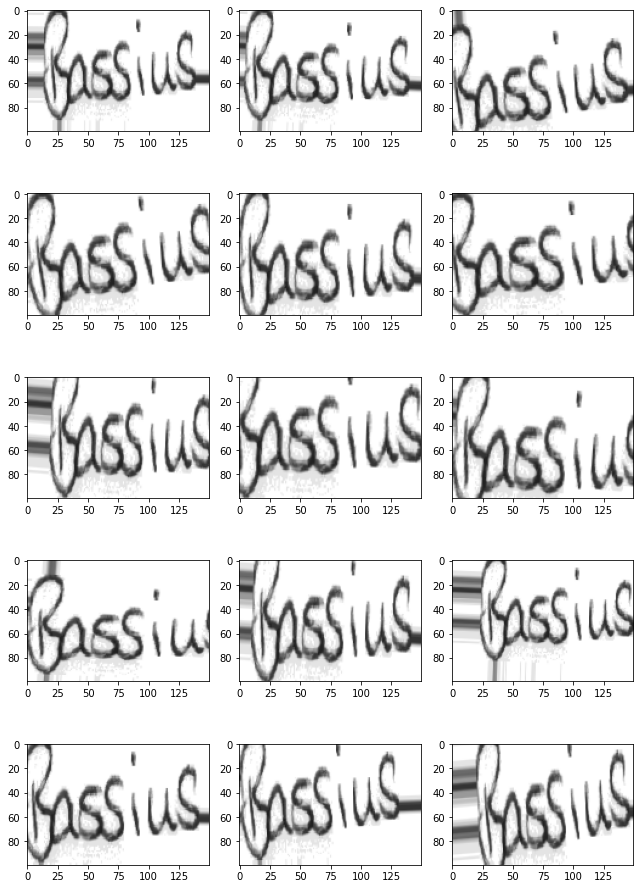
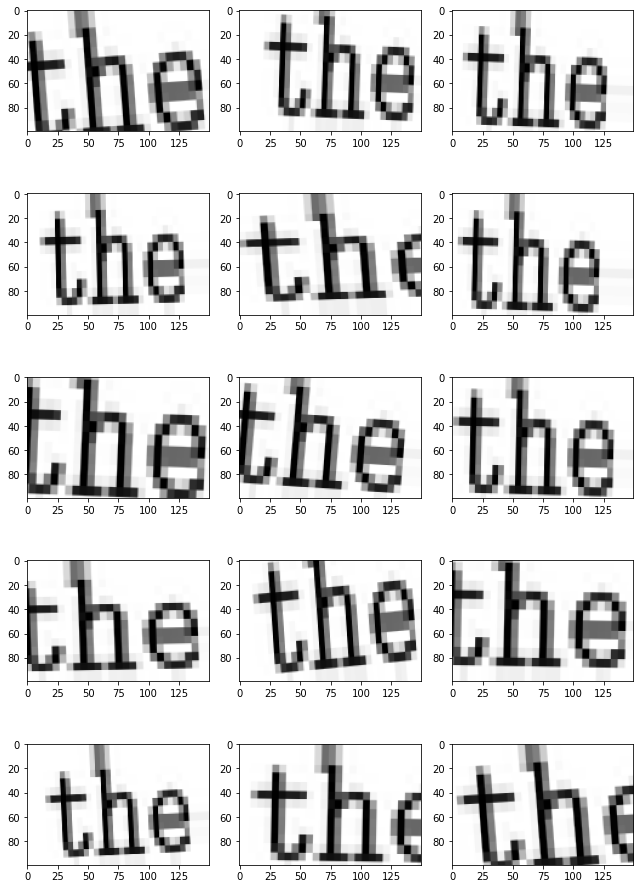
We split our dataset into two parts after random shuffling. We separated 20% of data for the validation set, and the rest 80% was assigned for training the model.

* 1. Data Augmentation

Having a large dataset is crucial for the performance of a deep learning model [10]. 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 of 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 10%

Figs 2 (a) and (b) showing some samples after data augmentation is applied for printed and handwritten words respectively.



**(a) (b)**

**Fig. 2.** Augmented samples for (a) printed (b) handwritten words

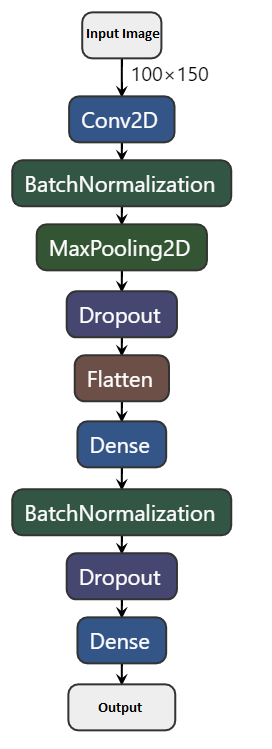
1. Work Methodology

As stated already, content type detection is a crucial part of developing an OCR system. A multitype OCR system needs to identify the type so that it can apply the type-specific layout segmentation, line segmentation, word segmentation models for accurately segmenting the content, and finally able to apply the correct recognition modules.

* 1. Proposed Model

Convolution, the basic building block of Convolutional Neural Network (CNN) [11], 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 uses CNN for the classification of document type identification to two classes: handwritten and printed. This model used convolution, max-pooling layer, fully connected dense layer, and regularization methods, such as batch normalization and dropout, as seen in Fig 3.



**Fig. 3.** Proposed CNN model

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 [12]. The second block is a Batch Normalization [13] 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% dropout layer and passed through a fully connected dense layer of 1 node with “Sigmoid” activation. This final layer is our output layer.

To minimize the error of the convolutional algorithms, optimization algorithms are heavily utilized. Our proposed model used Adam [14] 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 binary cross-entropy function.

* 1. 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 control when the training process stops.

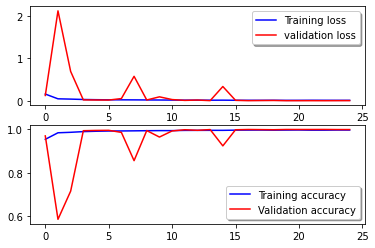
We set our initial epochs to 25, while the EarlyStop function monitored our test loss and stopped the training when needed. With this tool, our model stopped at 25 epochs, based on the model loss.

1. Performance

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

* 1. 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. 4 shows our learning curve for train test accuracy and loss.



**Fig. 4.** 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.

* 1. Accuracy and Loss

After 25 epochs, our model reached a maximum validation accuracy of 99.82%, with a minimum validation loss of 0.0048. For the training set, the maximum accuracy is 99.58%, with a minimum loss of 0.0122.

* 1. 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 and validation set are presented in Tables 2 and 3.

**Table 2.** Confusion Matrix for Train Set

|  |  |  |  |
| --- | --- | --- | --- |
| n = 45936 | Predicted printed | Predicted Handwritten |  |
| Actual printed | 23965 | 51 | 24,016 |
| Actual Handwritten | 8 | 21912 | 21,920 |
|  | 23,973 | 21,963 |  |

**Table 3.** Confusion Matrix for Validation Set

|  |  |  |  |
| --- | --- | --- | --- |
| n = 11485 | Predicted printed | Predicted Handwritten |  |
| Actual printed | 5,965 | 21 | 5,986 |
| Actual Handwritten | 1 | 5,498 | 5,499 |
|  | 5,966 | 5,519 |  |

1. Conclusion and Future Work

In this paper, we presented an effective model to identify handwritten and printed text in a mixed-type document. At the word-level, we achieved 99.82% validation accuracy. We have used a large word-level dataset to train our model and achieved 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 to that, in the proposed model, we can only detect two types—handwritten and printed. In the future, we plan to adapt this to a multiple-type (e.g. letterpress, or typewritten text) content detection problem.

References

1. “Bengali language - dialects & structure - mustgo.” [Online]. Available: https://www.mustgo.com/worldlanguages/bengali
2. G. C. P. in Bangladesh Language, “Bangla,” Feb 2020. [Online]. Available: https://planeta.com/bangla
3. “List of languages by number of native speakers,” May 2020. [Online]. Available: https://en.wikipedia.org/wiki/List of languages by number of native speakers
4. K. Zagoris, I. Pratikakis, A. Antonacopoulos, B. Gatos, and N. Papamarkos, “Distinction between handwritten and machine-printed text based on the bag of visual words model,” Pattern Recognition, vol. 47, no. 3, pp. 1051–1062, 2014.
5. E. Kavallieratou, S. Stamatatos, and H. Antonopoulou, “Machine-printed from handwritten text discrimination,” in Ninth International Workshop on Frontiers in Handwriting Recognition. IEEE, 2004, pp. 312–316.
6. C. Yam, C. Wolff, and C. Wolff, “Making sense of handwritten sections in scanned documents using the azure ml package for computer vision and azure cognitive services,” Mar 2020,http://devblogs.microsoft.com/cse/2018/05/07/handwriting-detection-and-recognition -in-scanned-documents-using-azure-ml-package-computer-vision-azure-cognitive-services-ocr.
7. 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.
8. 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)
9. Kaggle.com. 2020. Denoising Dirty Documents | Kaggle. [online] Available at: <https://www.kaggle.com/c/denoising-dirty-documents/data> [Accessed 29 May 2020].
10. 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
11. Albawi, Saad & Abed MOHAMMED, Tareq & ALZAWI, Saad. (2017). Understanding of a Convolutional Neural Network. 10.1109/ICEngTechnol.2017.8308186.
12. 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).
13. Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167.
14. Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.