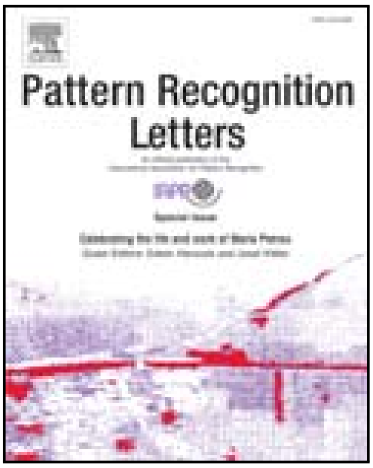


Handwritten Isolated Bangla Compound Character Recognition: a new benchmark using a novel deep learning approach

Saikat Roy, Nibaran Das, Mahantapas Kundu, Mita Nasipuri

PII: S0167-8655(17)30070-3
DOI: 10.1016/j.patrec.2017.03.004
Reference: PATREC 6759

To appear in: *Pattern Recognition Letters*

Received date: 18 December 2015
Revised date: 3 March 2017
Accepted date: 3 March 2017

Please cite this article as: Saikat Roy, Nibaran Das, Mahantapas Kundu, Mita Nasipuri, Handwritten Isolated Bangla Compound Character Recognition: a new benchmark using a novel deep learning approach, *Pattern Recognition Letters* (2017), doi: 10.1016/j.patrec.2017.03.004

Pattern Recognition Letters

Authorship Confirmation

Please save a copy of this file, complete and upload as the "Confirmation of Authorship" file.

As corresponding author I, Nibaran Das, hereby confirm on behalf of all authors that:

1. This manuscript, or a large part of it, has not been published, was not, and is not being submitted to any other journal.
2. If presented at or submitted to or published at a conference(s), the conference(s) is (are) identified and substantial justification for re-publication is presented below. A copy of conference paper(s) is(are) uploaded with the manuscript.
3. If the manuscript appears as a preprint anywhere on the web, e.g. arXiv, etc., it is identified below. The preprint should include a statement that the paper is under consideration at Pattern Recognition Letters.
4. All text and graphics, except for those marked with sources, are original works of the authors, and all necessary permissions for publication were secured prior to submission of the manuscript.
5. All authors each made a significant contribution to the research reported and have read and approved the submitted manuscript.

Signature Nibaran Das Date 4th March, 2017

List any pre-prints: N/A

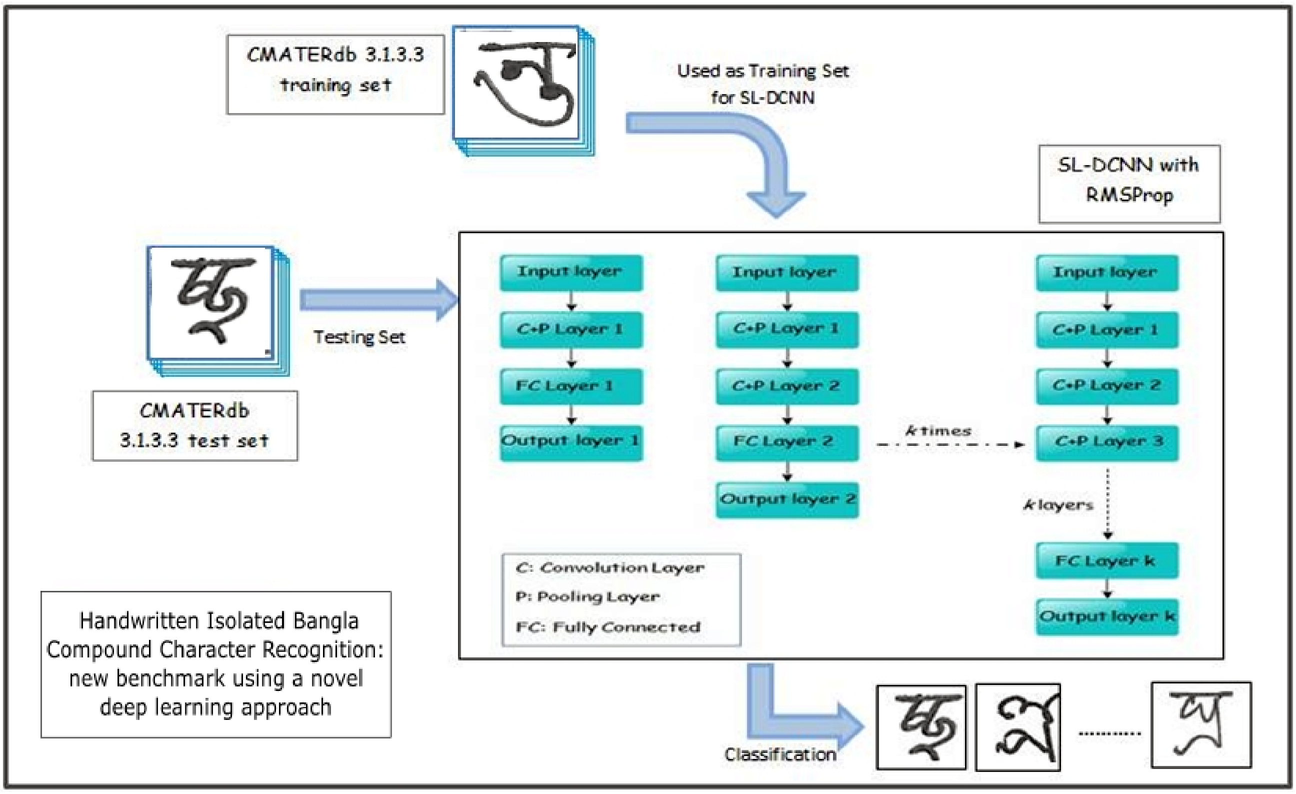
Relevant Conference publication(s) (submitted, accepted, or published): N/A

Justification for re-publication: N/A

Graphical Abstract (Optional)

Handwritten Isolated Bangla Compound Character Recognition: a new benchmark using a novel deep learning approach

Saikat Roy, Nibaran Das, Mita Nasipuri



Research Highlights (Required)

It should be short collection of bullet points that convey the core findings of the article. It should include 3 to 5 bullet points (maximum 85 characters, including spaces, per bullet point.)

- Introduction of deep learning techniques to handwritten *Bangla* compound character recognition
- Development of a novel supervised layerwise trained Deep Convolutional Neural Network
- Augmenting with RMSProp to achieve fast convergence and higher generalization
- Establishing a benchmark on the CMATERdb 3.1.3.3 *Bangla* compound character dataset
- A significant reduction of error rate from 19% to 9.67% on the dataset

ELSEVIER

Pattern Recognition Letters
journal homepage: www.elsevier.com

Handwritten Isolated Bangla Compound Character Recognition: a new benchmark using a novel deep learning approach

Saikat Roy[a], Nibaran Das[b,], Mahantapas Kundu[b], Mita Nasipuri[b]

[a]*Department of Information Technology, Jadavpur University, Kolkata, India*
[b]*Department of Computer Science & Engineering, Jadavpur University, Kolkata, India*

ABSTRACT

In this work, a novel deep learning technique for the recognition of handwritten *Bangla* isolated compound character is presented and a new benchmarkof recognition accuracy on the CMATERdb 3.1.3.3 dataset is reported. Greedy layer wise training of Deep Neural Network has helped to make significant strides in various pattern recognition problems. We employ *layerwise training* to Deep Convolutional Neural Networks (DCNN) in a supervised fashion and augment the training process with the RMSProp algorithm to achieve faster convergence. We compare results with those obtained from standard shallow learning methods with predefined features, as well as standard DCNNs. Supervised layerwise trained DCNNs are found to outperform standard shallow learning models such as Support Vector Machines as well as regular DCNNs of similar architecture by achieving error rate of 9.67% thereby setting a new benchmark on the CMATERdb 3.1.3.3 with recognition accuracy of 90.33%, representing an improvement of nearly 10%.

1. Introduction

Deep learning maybe loosely defined as an attempt to train a hierarchy of feature detectors – with each layer learning a higher representation of the preceding layer. The advent of deep learning has seen a resurgence in the use of (increasingly larger) neural networks (Szegedy et al. (2014)). Among many other areas of application of deep neural networks (DNNs), handwritten character recognition is one of the areas that has been extensively explored, particularly on the MNIST dataset (LeCun et al. (1998)).

But very few works have been published for Indian languages specially for *Bangla* which is sixth most popular language in the world (Bengali). *Bangla* language consists of 11 vowels, 39 consonants, 10 modifiers and 334 compound characters. Among them, compound characters are structurally complex and some of them resemble so closely (Figure 1) that the only sign of differences left between them are short straight lines, circular curves etc. (Das et al. (2014)) Due to this and the high number of classes, compound character recognition is a particularly challenging pattern recognition problem .

On the other hand, Deep learning has shown promise in recent years in multiple varied fields including handwritten character, speech and object recognition (Krizhevsky et al. (2012); Hinton et al. (2012); Lee et al. (2009); Goodfellow et al. (2013b)), natural language processing (He et al. (2014)) etc. Owing to its successful application in above areas, it is applied on the problem of handwritten *Bangla* compound character recognition which is much more challenging than MNIST dataset which consists of only 10 classes.

Deep learning methods usually benefit from a substantial amount of labeled or unlabeled data to build a powerful model to represent the data. The *Bangla* compound character datasets used in this work (Section 2.2), is freely downloadable for OCR research and consists of 171 unique classes with approximately 200 samples per class. The amount of data available, hence, necessitates the use of easily generalizable deep learning models or techniques to synthetically augment the data while training. Although shallow learning models, which have been previously used for *Bangla* compound character recognition, benefit from not having such requirements, the lower recognition accuracy produced by such models offset the benefits of not requiring a

[**] Corresponding author: Tel.: +91-332457-2407;
e-mail: nibaran@cse.jdvu.ac.in (Nibaran Das)

large amount of data.

Shallow learning models that are still popular for various tasks take an approach to pattern recognition which prioritizes the usage of *feature engineering* in extracting unique, robust and discriminating features. These features are then fed to machine learning models for classification or regression purposes. Over the last few decades various powerful multipurpose feature detectors have gained popularity such as HOG (Dalal and Triggs (2005)), SIFT (Lowe (2004); Surinta et al. (2015)), LBP (Ojala et al. (1996)), SURF (Bay et al. (2006)) etc. which have all been developed for a particular problem. But the major problem with such methodologies is that the usage of feature detectors need to be guided by the skill of the researchers using them.

Deep learning subverts the usage of handcrafted features by learning the features which are most effective for a particular problem. Inspired by Hubel and Wiesel's early work on the cat's visual cortex (Hubel and Wiesel (1959, 1965)), the convolutional neural network improved on the Neocognitron model (Fukushima (1980); Fukushima et al. (1983)). It was subsequently used in areas of handwriting recognition and more in the 80s and 90s (LeCun et al. (1989); LeCun and Bengio (1995); LeCun et al. (1990)) and showed promise for being used in deeper feedforward architectures. However, it was restricted by the limitations of computer hardware of the time. The LeNet model represented a culmination of sorts of work on CNNs in the 90s (LeCun et al. (1998)).

Although neural network models deeper than 4 layers had not been typically used till the mid of the first decade of the 2000s, the convolutional neural network had already been established as viable machine learning architecture (Simard et al. (2003); LeCun et al. (1998)). Subsequent years saw major works being done in the field of deep learning, particularly increasing usage of unsupervised training (Hinton and Salakhutdinov (2006)) and the greedy unsupervised layerwise training of DNNs (Bengio and Lamblin (2007)). The use of stacked denoising autoencoders (Vincent et al. (2008, 2010)), rectified linear activation units (Krizhevsky et al. (2012); He et al. (2015)) – which were resistant to the vanishing gradient problem (Hochreiter (1998)) – and the introduction of GPGPU Programming pushed the *depth* of neural networks (Szegedy et al. (2014)) even further.

During the last decade, the deep convolutional neural networks have carved out a large role in the discussion on deep learning. The popularization of Max Pooling (Scherer et al. (2010)), introduction of Dropout (Srivastava (2013)), Maxout networks (Goodfellow et al. (2013b); Cai and Liu (2016)) etc. led to widespread applications of DCNNs and DNNs in general. They generally encompassed character recognition, object detection, speech recognition, medical imaging amonng other pattern recognition problems (Ciresan et al. (2010); Ciresan and Schmidhuber (2013); Goodfellow et al. (2013b); Shin et al. (2016); Havaei et al. (2016); Jaderberg et al. (2016); Abdel-Zaher and Eldeib (2016); Ballester and Araujo (2016)). Various datasets such as MNIST, TIMIT, ImageNet, CIFAR and SVHN saw new benchmarks set on them using DNNs. With the availability of parallel programming on GPUs (Krizhevsky et al. (2012); Ciresan et al. (2011); Li et al. (2016)), interest in the development of architectures deeper than 15 layers (Simonyan and Zisserman (2014); Sercu et al. (2016)) and transferring the experience of models trained for one domain to another (Holder et al. (2016); Shouno et al. (2015)), the explosion of interest in deep learning research is a well established fact.

2. Previous work

2.1. Literature of Handwritten Bangla Compound Character Recognition

Ample works on handwritten *Bangla* character recognition have been reported through the last decade. There are various research works which have attempted to deal with the history of *Bangla* character recognition (Pal and Chaudhuri (2004); Basu et al. (2005, 2009a); Das et al. (2009, 2010); Pal et al. (2012); Bag and Harit (2013)). However, the collection of literature on handwritten *Bangla* compound character recognition is far shallower as the majority of the literature focuses only on *Bangla* numeral and basic character recognition.

The work on handwritten *Bangla* compound characters is relatively newer and started during the last decade. Handwritten compound characters bring an added variability to the characters and make for a challenging pattern recognition problem. In (Pal et al. (2007)), features extracted from the directional information of the arc tangent of the gradient in combination with a Modified Quadratic Discriminant Function classifier were used for handwritten compound character recognition on a problem size of 138 classes of compound characters. In another work (Das et al. (2009)), Quad tree based features were used for recognition of 55 frequently occurred compound characters covering 90% of the total of compound characters in *Bangla* using an Multilayer Perceptron (MLP) classifier.In another work of the same author (Das et al. (2010)), 50 Basic and 43 frequently used Compound characters are considered to create a 93 class character recognition problem which was solved using shadow and quad tree based longest run features and MLP and support vector machine as classifiers. In (Bag et al. (2011)), a method to improve classification performance on *Bangla* Basic characters using topological features derived from the convex shapes of various strokes was proposed.

More recently, (Das et al. (2015)) describes handwritten *Bangla* Character recognition using a soft computing paradigm embedded in a two pass approach. More specifically highly misclassified classes were combined to form a single group in the first pass or coarse classification. In the second pass, group specific local features were identified using Genetic Algorithm based region selection strategy to classify the appropriate class from the groups formed in the earlier pass. They used two different sets of features – a) convex hull based features b) Longest run based features with Support Vector Machines (SVM), a well known classifier for this purpose. They reported a recognition accuracy of 87.26% on a dataset of handwitten *Bangla* characters consisting of Basic characters, Compound characters and Modifiers. In another work (Sarkhel et al. (2016)) , Sarkhel et al approached the issue from a perspective of multi-objective based region selection problem where the most informative regions of character samples were used

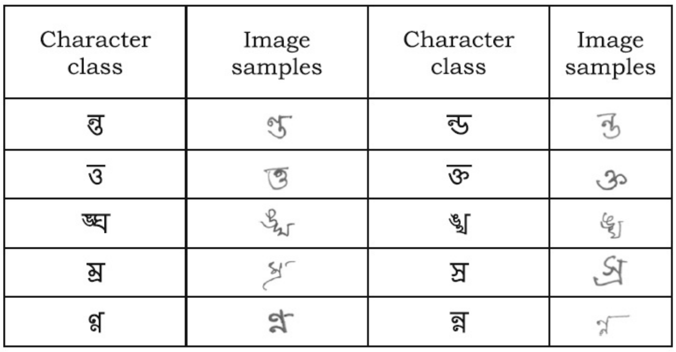

Character class
Image samples
Character class
Image samples
ন্তু
ন্ড
ত্ত
ঙ্ক
ঙ্ঘ
ঙ্খ
ম্র
স্র
প্ন
ন্ন

Fig. 1: Random samples from a few closely resembling classes (Das et al. (2014))

Table 1: Major differences between MNIST and CMATERdb 3.1.3.3

Parameter	MNIST	CMATERdb 3.1.3.3
Scale	Uniform	Non-uniform
Translation	Centred	Non-Centred
Problem Size	10 class	171 class
Training Data	60000 samples	34439 samples
Testing Data	10000 samples	8520 samples

to train SVM classifiers for character recognition. Two algorithms for optimization, specifically a Non-dominated Sorting Harmony Search Algorithm and Non-dominated Sorting Genetic Algorithm II were used to select the most informative regions with the objective of minimal recognition cost and maximal recognition accuracy. A recognition accuracy of 86.65% on a mixed dataset of *Bangla* numerals, basic and compound characters was reported. In Section 2.3, we discuss another methodology on handwritten *Bangla* compound character recognition on CMATERdb 3.1.3.3, a standard freely available handwritten *Bangla* compound character dataset.

2.2. *Description of CMATERdb 3.1.3.3: Isolated Handwritten Bangla Compound Character Database*

The CMATERdb 3.1.3.3 is a database having 171 unique classes of isolated grayscale images of *Bangla* compound characters. There are 34439 individual training samples and 8520 test samples in the dataset. The images in the dataset, being neither centered nor of uniform scale, constitute a difficult pattern recognition problem.

For ease of comparison, Table 1 presents the fundamental characteristics of the widely used MNIST dataset and the CMATERdb 3.1.3.3 dataset. It is apparent that a higher number of classes, unnormalized images in terms of scale and translation, and with a lower amount of training data, the basic pattern recognition problem with CMATERdb 3.1.3.3 is more complicated than that in the MNIST dataset.

2.3. *Methodology of previous benchmark*

The previous benchmark (Das et al. (2014)) (which happens to be the original benchmark) on the CMATERdb 3.1.3.3 isolated compound character database used the traditional methodology of feature engineering in combination with a classifier. In particular, a feature descriptor composed of convex hull and Quad Tree based features was used with an SVM classifier.

For the Quad Tree based features, a quad tree of depth 2 was created where the partitioning of the image was done at the centre of gravity of the black pixels. This recursive partitioning resulted in 21 sub-images representing the nodes of the Quad Tree. From these sub-images, row-wise, column-wise and diagonal-wise longest run features (Basu et al. (2009c)) are extracted. For the convex hull based features, multiple 'bay' and 'lake' descriptors are defined based on the convex hull around the black character pixels (Das et al. (2010)). The combined features are used to train a Support Vector Machine for character recognition.

An error rate of approximately 19% was achieved as the erstwhile benchmark on the 171 class dataset.

3. Proposed Recognition Strategy

3.1. *Deep Convolutional Neural Network*

The simplest description of a DCNN would be to imagine the LeNet-5 Convolutional Neural Network (LeCun et al. (1998)) with an arbitrarily high number of Convolution and Pooling layers. The architecture of a DCNN consists of possibly alternate Convolution or Pooling layers, before ending up in one or many fully connected layers and an output layer. Concepts of shared weights, local receptive fields and subsampling that are essential to a DCNN have been discussed repeatedly in literature (LeCun et al. (1989, 1998); Ciresan et al. (2011); Bengio (2009)) and have not been repeated here.

3.2. *Supervised Layerwise trained Deep Convolutional Neural Networks*

For supervised learning using any particular DCNN architecture, to get the most out of the model, we propose an alternative to straight forward training on the entire model architecture. Greedy Unsupervised Layerwise (Bengio and Lamblin (2007)) training as applied to models like denoising autoencoders suggest training the neural network layer by layer and then a fine tuning of the network. A supervised variant of the unsupervised layerwise training methods, made popular in literature in the last decade, is put forward as a superior training method for DCNNs without any significant change to model architecture or training algorithms.

Supervised layerwise training of a Deep Convolutional Neural Network (SL-DCNN) essentially involves training a DCNN by building the model incrementally by adding Convolution and Pooling layers one by one and training them before adding more layers. After the entire model is built, a fine tuning is performed by training the entire model at a very low learning rate for a short number of iterations. In fact, our method has the same 2 steps that are common to unsupervised layerwise training (Vincent et al. (2010)) that is, layerwise training and fine tuning. The only difference is that the cost function for unsupervised training has been replaced with a cost function for supervised

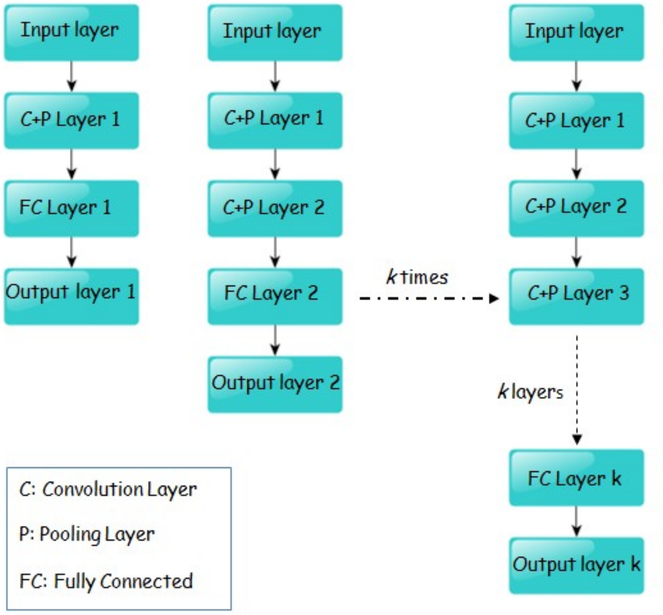


Fig. 2: Block diagram of an SL-DCNN layered k times

training based on classification error. Figure 2 illustrates an SL-DCNN layered k times.

Since there is no obvious way of reusing the output layer and the fully connected layer, we choose to retrain a new of each of the above with every new layer added. The reason for doing so is that the use of at least one fully connected layer seems to stabilize the learning process when added. Hence, with the computational cost of adding the 2 layers being minimal, a fully connected layer is added every time a new convolutional and pooling layer is added. The old fully connected and output layers are obviously removed before the addition of new ones.

In (deep) convolutional neural networks, the relatively low fan-in of individual neurons coupled with the use of Rectified Linear activation units do not allow the gradient to diffuse much. However, issues such as convergence to local minima as well as lower generalization performance do persist and layerwise training has been shown to be beneficial (Erhan et al. (2010)) when applied to such problems. The same is observed when supervised layerwise training is applied by us to DCNN training.

While there are no architectural differences between the DCNN and SL-DCNN, the method of realization and training of the complete model is where the contrast lies. Layerwise training enables an SL-DCNN to be have an iterative increase in model complexity. However, in case of a DCNN, it involves training the complete model at once. The training methodology of an SL-DCNN demonstrates distinctly faster convergence and higher generalization as compared to the comparatively simple DCNN.

3.3. RMSProp

The concept of local learning rates were introduced to avoid a flat global learning rate and enable faster training and better convergence – all of which are desirable in deep learning systems.

```
Data: trn, tst, lrate1, lrate2
Result: A trained SL-DCNN model
begin
    $trn \longleftarrow$ Training Data;
    $tst \longleftarrow$ Testing Data;
    $INPUT \longleftarrow$ Layer specifying dimensions of image;
    $CP \longleftarrow$ Convolutional and Pooling layer;
    $FC \longleftarrow$ Fully Connected layer;
    $OUTPUT \longleftarrow$ Specifies the number of output classes;
    $lrate1 \longleftarrow$ Comparatively High Learning Rate;
    $lrate2 \longleftarrow$ Low Learning Rate;
    // model is initialized as null
    $mdl \longleftarrow \phi$ ;
    AddLayer($mdl, INPUT$);
    AddLayer($mdl, CP$);
    AddLayer($mdl, FC$);
    AddLayer($mdl, OUTPUT$);
    // RMSProp is used when training model
    TrainModel($mdl, trn, tst, lrate1$);
    // index here points to last layer
    $index \longleftarrow -1$;
    while not all layers have been added do
        RemoveLayer (mdl, $index$);
        RemoveLayer (mdl, $index$);
        AddLayer($mdl, CP$);
        AddLayer($mdl, FC$);
        AddLayer($mdl, OUTPUT$);
        TrainModel($mdl, trn, tst, lrate1$);
    end
    TrainModel($mdl, trn, tst, lrate2$);
end
```

Algorithm 1: Technique for training an SL-DCNN

RMSProp is a technique conceptualized by Hinton et al (G.Hinton, slide 6) for adaptive local learning rates. It is essentially, a customized version for mini-batch gradient of another adaptive learning rate method called rprop (Riedmiller and Braun (1993)), which was designed for stochastic gradient descent.

The method works by keeping a moving average of the squared gradient for each weight and dividing the gradient by the square root of this value. RMSProp has been recently discussed in literature (Dauphin et al. (2015); Carlson et al. (2015)) and has been shown to significantly hasten convergence despite possessing certain limitations. The learning rate adjustment parameter at time (or iteration t) for each weight r_t is given by,

$$r_t = (1 - \gamma) f'(\theta_t)^2 + \gamma r_{t-1} \tag{1}$$

where γ is the decay rate, $f'(\theta_t)$ is the derivative of the error with respect to the weight at time (or iteration) t and and r_{t-1} is the previous value of the adjustment parameter.

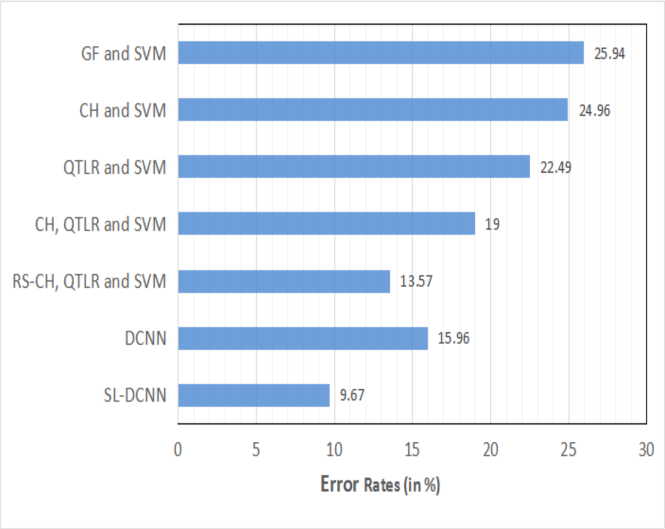


Fig. 3: Error rates of the different models on the CMATERdb 3.1.3.3. CH, QTLR and SVM represents the old benchmark on the dataset.

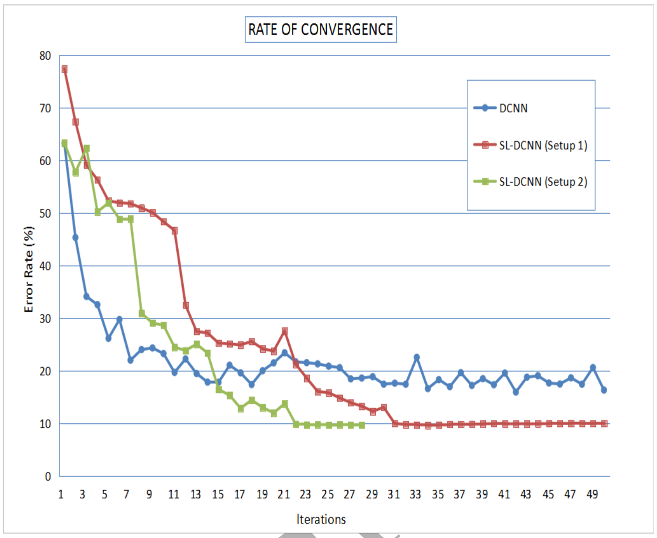


Fig. 4: Rates of Convergence of DCNN and SL-DCNN (Setup 1 and 2). Setup 2 error rates are nearly equivalent to Setup 1 error rates

The update to the weights is then given by,

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{r_t}} f'(\theta_t) \tag{2}$$

where α is the global learning rate, and θ_t and θ_{t+1} are the values of the weight at time (or iteration) t and $t+1$ respectively.

3.4. SL-DCNN with RMSProp

RMSProp has been used generously with SL-DCNN thus speeding up layerwise training as well as fine tuning. Unlike unsupervised layerwise models, supervised layerwise models have no way of utilizing vast amounts of unlabeled data. However, it maybe said without exaggeration that an SL-DCNN model combined with RMSProp represents the most efficient way of utilizing labeled data. Algorithm 1 demonstrates the method for building an SL-DCNN layer by layer.

4. Experiments

A series of experiments were performed to compare the convergence rates of the SL-DCNN and the DCNN, as well as to establish a new benchmark on the CMATERdb 3.1.3.3. The experiments were run on a system having an Intel Core i3 processor, 4GB RAM, with Ubuntu 14.04 operating system. The models were coded in Python and used the Pylearn2 machine learning framework (Goodfellow et al. (2013a)) and dependencies such as Theano (Theano Development Team (2016)), NumPy and SciPy. Pillow (fork of PIL) was used for basic image processing tasks while HDF5 (h5py) was used for handling dataset storage issues.

4.1. Preprocessing

Due to the nature of the dataset and the learning model used, the preprocessing on the dataset was kept to a minimum. The data was subjected to standardization only and resized to a width and height of 70 pixels to make it convenient for the DCNN.

4.2. Description of DCNN and SL-DCNN architectures

Owing to the difficulty of hyper-parameter search in DNNs, a uniform architecture was used for both types of models, with a standard 4x4 filter size in all layers. The fully connected layer in all models consist of 1500 units. The convention used throughout the article to describe architectures is elaborated below.

- **xCy:** A convolution layer with x number of filters and a filter size of yXy.
- **xPy:** A pooling layer with a pool size of xXx and a pool stride of y.
- **xFC:** A fully connected layer of x number of hidden units.
- **xSM:** A softmax output layer of x number of output units.

Based on the above convention the architecture used can be written as 64C4-4P2-64C4-4P2-64C4-4P2-1500FC-171SM. It is worth mentioning that for regular DCNNs, the architecture is trained by taking all layers at once, while the SL-DCNN is trained in a layerwise fashion.

4.3. Learning Process Details

The techniques, mentioned below, were kept consistent across the various DNN experiments to aid the learning process.

1. Mini-batch gradient descent: Mini-batch gradient descent with a batch size of 100 was used for all our experiments.
2. Rectified Linear Units: Rectified Linear Activation Units were used as the activation units for the neurons in all neural network models.
3. Softmax Output Units: Softmax output units were used in the output layers of all the models.

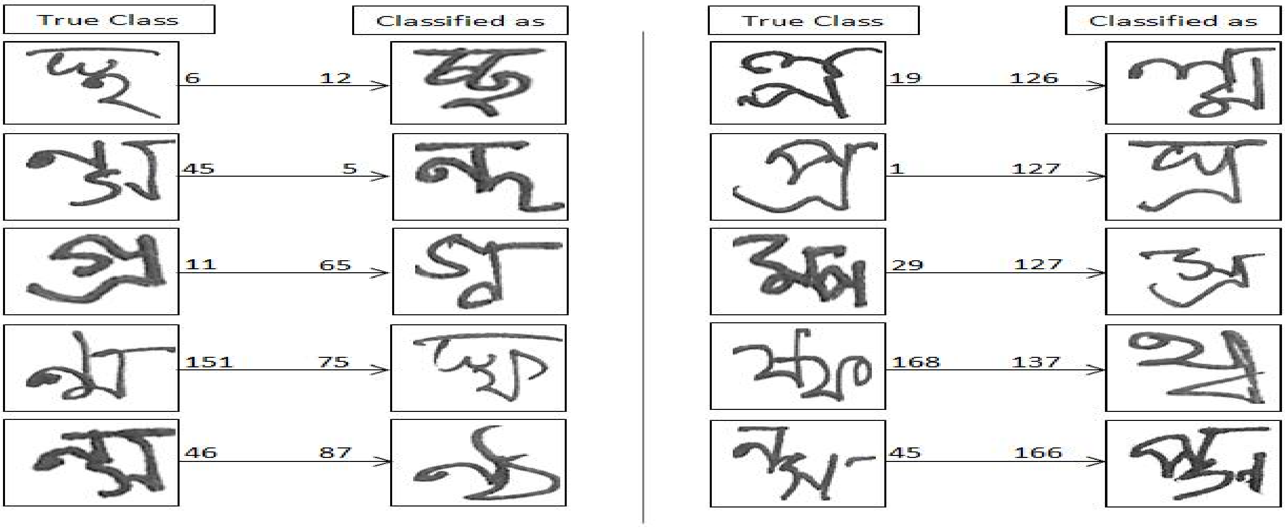


Fig. 5: A sample of frequently misclassified classes by the final SL-DCNN model

4. Decaying Learning Rate: In our experiments, the learning rate was made to decay from the original value to 0.1 times of the original value during a total of 50 iterations.
5. Shuffled Datasets: Shuffling of training examples has been shown to benefit training of neural networks (LeCun et al. (2012)) and was used in all our experiments.
6. RMSProp: The RMSProp algorithm as discussed earlier was used in all experiments performed as it significantly expedited the learning.

4.4. Experimental Setup

Two different experimental setups are designed by using different number of iterations per layer for the SL-DCNN for analytical purposes.

1. **Setup 1:** Each Convolution and Pooling layer of the SL-DCNN is trained for 10 iterations, and after addition of 3 Convolution and Pooling layers, a fine tuning is performed for 20 iterations at a very low learning rate. This results in a total of 50 iterations – 30 for 3 Convolution and Pooling layers and 20 for the fine tuning. This is the same as the number of iterations used by the DCNN. This setup is used primarily to establish a benchmark on the CMATERdb 3.1.3.3 by placing both the DCNN and SL-DCNN on an equal footing.
2. **Setup 2:** Each Convolution and Pooling layer of the SL-DCNN is trained for 7 iterations, and after addition of 3 Convolution and Pooling layers, a fine tuning is performed for 7 iterations. This leads to an SL-DCNN model trained to nearly half as many iterations (to be precise 28) as the DCNN model. The purpose of this setup is to illustrate the rate of convergence of the SL-DCNN.

5. Results and Analysis

5.1. Benchmark on the CMATERdb 3.1.3.3.

One of the major aims of this work was to achieve a new benchmark on the CMATERdb 3.1.3.3. As described in Section 2.3, the previous benchmark on the dataset was around 19% error rate (CH, QTLR and SVM – Das et al. (2014)). Additionally, we also provide reference results to alternative shallow learning based techniques, each used to train an SVM classifier — Quad Tree Longest Run features (QTLR and SVM – Das et al. (2009); Wen et al. (2007)), Gradient features (GF and SVM – Basu et al. (2009b)), Convex Hull features (CH and SVM – Das et al. (2010)), Region Sampled Convex Hull and Quad Tree Longest Run features (RS-CH,QTLR and SVM - Sarkhel et al. (2016)) — for classifying the CMATERdb 3.1.3.3 isolated compound character dataset. We also compare with a regular DCNN model with similar architecture. Our present work surpasses the previous benchmark with our SL-DCNN model providing an error rate of 9.67% compared to 15.96% for the regular DCNN.

Figure 3 illustrates the final error rates on the various models mentioned previously, with the DCNN and SL-DCNN models following parameters mentioned in Setup 1.

5.2. Rate of Convergence

The rate of convergence is a compelling argument as to why SL-DCNN as a model represents a highly efficient deployment of a convolutional neural network. With no change in architecture and a change in the training methodology, significant gains are achieved using the SL-DCNN. To make this self-evident, 2 different experimental setups are employed for the SL-DCNN as previously stated.

Figure 4 illustrates the rate of convergence for DCNN and SL-DCNN (Setup 1 and Setup 2) . Setup 1 demonstrates that the SL-DCNN converges faster and to a higher accuracy than a DCNN. We have also observed that with the usage of RMSProp as a component for the layerwise training, the models can converge to equivalent accuracies at lower number of iterations. To illustrate this, we have used Setup 2.

Setup 2 is however designed to make the SL-DCNN run for nearly half the total number of iterations as in Setup 1. It was seen in our experiments with Setup 2, that an error rate of 9.69% was achieved which is nearly equivalent to the error rate of

9.67% in Setup 1. The model in Setup 2 was seen to converge at iteration 25. This demonstrates that the SL-DCNN trained at each layer to a very low number of iterations still gives a comparatively low error rate. Hence, it can be said that the SL-DCNN model possesses a fast rate of convergence.

5.3. Analysis of Misclassification

Although providing a significant reduction in error rates, there are still difficulties inherent to the dataset that the benchmark model struggled with. Inspection of the structure of the characters, along with an analysis of misclassifications reveals some of the issues which makes *Bangla* compound character recognition a particularly challenging pattern recognition problem. Figure 5 represents some samples of frequent misclassifications by the final SL-DCNN model. Two reasons for such misclassifications are apparent:

1. As mentioned previously, there exist some *Bangla* coumpound character classes with having high structural similarities. One type of such similarities are those found in pairs of compound characters such as 45 and 5 or 46 and 87 (shown in Figure 5), which share one common basic character for true class and misclassified class, hence implicitly showing structural similarity.
2. Also, there are pairs such as 168 and 137 or 6 and 12 (also shown in Figure 5) which, despite not sharing any basic *Bangla* character, are written in a manner where they share major structural constructs.

6. Conclusion

DCNNs in general, regardless of the hardware available, are costly to train but extremely effective machine learning models. Hence, fast convergence should be viewed as an important front of research. Having said that, SL-DCNNs represents such a model with higher generalization and faster convergence compared to regular DCNNs.

As a machine learning model, SL-DCNN shows promise for use in allied areas of research and the exploration of the same remains an issue that maybe explored in future works. Also, there remains the possibility to augment the ability of SL-DCNNs by leveraging recent advancements in the area of deep learning such as Dropout, Maxout units, *significantly deeper* networks and Transfer Learning.

However, the particular SL-DCNN architecture used in this work has been powerful enough to set a new benchmark of 9.67% error rate on the rather difficult CMATERdb 3.1.3.3 handwritten *Bangla* isolated compound character dataset. This represents a lowering of error rates by nearly 10% from previously set benchmarks on the same and is an excellent result in the area of *Bangla* compound character recognition.

Acknowledgments

This work has been supported by the *Center for Microprocessor Applications for Training, Education and Research (CMATER)* laboratory at the Department of Computer Science & Engineering of Jadavpur University, Kolkata. The resources provided by the lab have been invaluable in helping us progress with our research.

References

Abdel-Zaher, A.M., Eldeib, A.M., 2016. Breast cancer classification using deep belief networks. Expert Systems with Applications 46, 139–144.

Bag, S., Bhowmick, P., Harit, G., 2011. Recognition of bengali handwritten characters using skeletal convexity and dynamic programming, in: Emerging Applications of Information Technology (EAIT), 2011 Second International Conference on, pp. 265–268.

Bag, S., Harit, G., 2013. A survey on optical character recognition for bangla and devanagari scripts. Sadhana .

Ballester, P., Araujo, R.M., 2016. On the performance of googlenet and alexnet applied to sketches, in: Thirtieth AAAI Conference on Artificial Intelligence.

Basu, S., Das, N., Sarkar, R., Kundu, M., Nasipuri, M., Basu, D.K., 2005. Handwritten bangla alphabet recognition using an mlp based classifier, in: 2nd National Conf. on Computer Processing of Bangla-2005, , Dhaka, Bangladesh, arXiv preprint arXiv:1203.0882. pp. pp–285.

Basu, S., Das, N., Sarkar, R., Kundu, M., Nasipuri, M., Basu, D.K., 2009a. A hierarchical approach to recognition of handwritten Bangla characters. Pattern Recognition 42, 1467–1484.

Basu, S., Das, N., Sarkar, R., Kundu, M., Nasipuri, M., Basu, D.K., 2009b. A hierarchical approach to recognition of handwritten bangla characters. Pattern Recognition 42, 1467–1484.

Basu, S., Das, N., Sarkar, R., Kundu, M., Nasipuri, M., Basu, D.K., 2009c. Recognition of numeric postal codes from multi-script postal address blocks, in: Proceedings of the 3rd International Conference on Pattern Recognition and Machine Intelligence, Springer-Verlag, Berlin, Heidelberg. pp. 381–386.

Bay, H., Tuytelaars, T., Van Gool, L., 2006. Surf: Speeded up robust features, in: European Conference on Computer Vision. volume 3951 of *Lecture Notes in Computer Science*, pp. 404–417.

Bengali, . Summary by language size. https://www.ethnologue.com/statistics/size. Accessed: 2017-03-04.

Bengio, Y., 2009. Learning Deep Architectures for AI. Foundations and Trends in Machine Learning 2, 1–127.

Bengio, Y., Lamblin, P., 2007. Greedy layer-wise training of deep networks. Advances in Neural Information Processing Systems , 153–160.

Cai, M., Liu, J., 2016. Maxout neurons for deep convolutional and lstm neural networks in speech recognition. Speech Communication 77, 53–64.

Carlson, D., Cevher, V., Carin, L., 2015. Stochastic spectral descent for restricted boltzmann machines, in: Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, pp. 111–119.

Ciresan, D., Meier, U., Masci, J., 2011. Flexible, high performance convolutional neural networks for image classification. International Joint Conference on Artificial Intelligence , 1237–1242.

Ciresan, D., Schmidhuber, J., 2013. Multi-Column Deep Neural Networks for Offline Handwritten Chinese Character Classification.

Ciresan, D.C., Meier, U., Gambardella, L.M., Schmidhuber, J., 2010. Deep, big, simple neural nets for handwritten digit recognition. Neural computation 22, 3207–3220. arXiv:1003.0358.

Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), pp. 886–893.

Das, N., Acharya, K., Sarkar, R., Basu, S., Kundu, M., Nasipuri, M., 2014. A benchmark image database of isolated Bangla handwritten compound characters. International Journal on Document Analysis and Recognition (IJDAR) 17, 413–431.

Das, N., Basu, S., Sarkar, R., Kundu, M., Nasipuri, M., Basu, D., 2009. Handwritten bangla compound character recognition: Potential challenges and probable solution, in: IICAI, pp. 1901–1913.

Das, N., Das, B., Sarkar, R., Basu, S., Kundu, M., Nasipuri, M., 2010. Handwritten bangla basic and compound character recognition using MLP and SVM classifier. CoRR abs/1002.4040.

Das, N., Sarkar, R., Basu, S., Saha, P.K., Kundu, M., Nasipuri, M., 2015. Handwritten bangla character recognition using a soft computing paradigm embedded in two pass approach. Pattern Recognition 48, 2054 – 2071.

Dauphin, Y.N., de Vries, H., Chung, J., Bengio, Y., 2015. Rmsprop and equilibrated adaptive learning rates for non-convex optimization. CoRR .

Erhan, D., Bengio, Y., Courville, A., Manzagol, P.A., Vincent, P., Bengio, S., 2010. Why does unsupervised pre-training help deep learning? J. Mach. Learn. Res. 11, 625–660.

Fukushima, K., 1980. Neocognitron: a self organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological cybernetics 36, 193–202.

Fukushima, K., Miyake, S., Ito, T., 1983. Neocognitron: A neural network model for a mechanism of visual pattern recognition. IEEE Transactions On Systems Man And Cybernetics SMC-13, 826–834.

G.Hinton, slide 6, . Lecture slide 6 of geoffrey hinton's course. http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf. Accessed: 2015-05-23.

Goodfellow, I.J., Warde-Farley, D., Lamblin, P., Dumoulin, V., Mirza, M., Pascanu, R., Bergstra, J., Bastien, F., Bengio, Y., 2013a. Pylearn2: a machine learning research library. arXiv preprint arXiv:1308.4214 .

Goodfellow, I.J., Warde-Farley, D., Mirza, M., Courville, A., Bengio, Y., 2013b. Maxout Networks. arXiv preprint , 1319–1327.

Havaei, M., Davy, A., Warde-Farley, D., Biard, A., Courville, A., Bengio, Y., Pal, C., Jodoin, P.M., Larochelle, H., 2016. Brain tumor segmentation with deep neural networks. Medical Image Analysis .

He, K., Zhang, X., Ren, S., Sun, J., 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 1026–1034.

He, X., Gao, J., Deng, L., 2014. Deep learning for natural language processing: Theory and practice (tutorial), CIKM.

Hinton, G., Deng, L., Yu, D., Dahl, G., rahman Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T., Kingsbury, B., 2012. Deep neural networks for acoustic modeling in speech recognition. Signal Processing Magazine .

Hinton, G.E., Salakhutdinov, R.R., 2006. Reducing the dimensionality of data with neural networks. Science 313, 504–507.

Hochreiter, S., 1998. The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions.

Holder, C.J., Breckon, T.P., Wei, X., 2016. From on-road to off: Transfer learning within a deep convolutional neural network for segmentation and classification of off-road scenes, in: European Conference on Computer Vision, Springer. pp. 149–162.

Hubel, D.H., Wiesel, T.N., 1959. Receptive fields of single neurones in the cat's striate cortex. The Journal of physiology 148, 574–591.

Hubel, D.H., Wiesel, T.N., 1965. Receptive Fields and Functional Architecture in two nonstriate visual areas (18 and 19) of the cat. Journal of Neurophysiology 28, 229–289.

Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A., 2016. Reading text in the wild with convolutional neural networks. International Journal of Computer Vision 116, 1–20.

Krizhevsky, A., Sutskever, I., Hinton, G., 2012. Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems , 1–9.

LeCun, Y., Bengio, Y., 1995. Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks 3361, 255–258.

LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D., 1989. Backpropagation Applied to Handwritten Zip Code Recognition.

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient-based learning applied to document recognition. Proceedings of the IEEE 86, 2278–2323.

LeCun, Y., Bottou, L., Orr, G.B., Müller, K.R., 2012. Efficient backprop. Lecture Notes in Computer Science , 9–48.

LeCun, Y., Jackel, L.D., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., 1990. Handwritten Digit Recognition with a Back-Propagation Network, in: Advances in Neural Information Processing Systems, pp. 396–404.

Lee, H., Pham, P., Largman, Y., Ng, A.Y., 2009. Unsupervised feature learning for audio classification using convolutional deep belief networks, in: Advances in Neural Information Processing Systems 22, pp. 1096–1104.

Li, C., Yang, Y., Feng, M., Chakradhar, S., Zhou, H., 2016. Optimizing memory efficiency for deep convolutional neural networks on gpus. arXiv preprint arXiv:1610.03618 .

Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60.

Ojala, T., Pietikinen, M., Harwood, D., 1996. A comparative study of texture measures with classification based on featured distributions. Pattern Recognition 29, 51 – 59.

Pal, U., Chaudhuri, B., 2004. Indian script character recognition: A survey. Pattern Recognition 37, 1887 – 1899.

Pal, U., Jayadevan, R., Sharma, N., 2012. Handwriting recognition in indian regional scripts: A survey of offline techniques. ACM Transactions on Asian Language Information Processing 11, 1:1–1:35.

Pal, U., Wakabayashi, T., Kimura, F., 2007. Handwritten bangla compound character recognition using gradient feature, in: Information Technology, (ICIT 2007). 10th International Conference on, pp. 208–213.

Riedmiller, M., Braun, H., 1993. A direct adaptive method for faster backpropagation learning: the RPROP algorithm. IEEE International Conference on Neural Networks .

Sarkhel, R., Das, N., Saha, A.K., Nasipuri, M., 2016. A multi-objective approach towards cost effective isolated handwritten bangla character and digit recognition. Pattern Recognition 58, 172–189.

Scherer, D., Müller, A., Behnke, S., 2010. Evaluation of pooling operations in convolutional architectures for object recognition, in: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 92–101.

Sercu, T., Puhrsch, C., Kingsbury, B., LeCun, Y., 2016. Very deep multilingual convolutional neural networks for lvcsr, in: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE. pp. 4955–4959.

Shin, H.C., Roth, H.R., Gao, M., Lu, L., Xu, Z., Nogues, I., Yao, J., Mollura, D., Summers, R.M., 2016. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. IEEE Transactions on Medical Imaging 35, 1285–1298.

Shouno, H., Suzuki, S., Kido, S., 2015. A transfer learning method with deep convolutional neural network for diffuse lung disease classification, in: International Conference on Neural Information Processing, Springer. pp. 199–207.

Simard, P., Steinkraus, D., Platt, J., 2003. Best practices for convolutional neural networks applied to visual document analysis. Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings. .

Simonyan, K., Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556.

Srivastava, N., 2013. Improving neural networks with dropout. Thesis .

Surinta, O., Karaaba, M.F., Schomaker, L.R., Wiering, M.A., 2015. Recognition of handwritten characters using local gradient feature descriptors. Engineering Applications of Artificial Intelligence 45, 405 – 414.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., 2014. Going deeper with convolutions. CoRR abs/1409.4842.

Theano Development Team, 2016. Theano: A Python framework for fast computation of mathematical expressions. arXiv e-prints abs/1605.02688.

Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A., 2008. Extracting and composing robust features with denoising autoencoders. Proceedings of the 25th international conference on Machine learning - ICML '08 , 1096–1103.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A., 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of Machine Learning Research 11, 3371–3408.

Wen, Y., Lu, Y., Shi, P., 2007. Handwritten bangla numeral recognition system and its application to postal automation. Pattern recognition 40, 99–107.