Journal of Information Systems Engineering and Management
2025, 10(16s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Enhanced Handwritten Digit Recognition with a Hybrid Optimization Framework for Deep Learning Techniques

Neha Bhatia[1], Manjula Shanbhog[2], Sandeep Kumar[3], K. Anu Radha[4]
[1] Research Scholar, SRU, Alwar & Assistant Professor, Department of Information Technology, Panipat Institute of Engineering & Technology, Panipat, HR, India.
Mail id: neha.20july@gmail.com
[2]Associate Professor, Department of Computer Science and Engineering, Christ Deemed to be) University, Ghaziabad, UP, India.
Mail id: manjulashanbhog@gmail.com
[3]Assistant Professor, Department of Information Technology, Panipat Institute of Engineering & Technology, Panipat, HR, India.
Mail id:sandeepgoel1986@gmail.com
[4]Associate Professor, Department of AI&DS,
KL University, Guntur, AP, India.
Mail id: anuradhakonidena@kluniversity.in

ARTICLE INFO

Received: 05 Dec 2024

Revised: 28 Jan 2025

Accepted: 06 Feb 2025

ABSTRACT

One of the interesting applications for machine deep learning in the arena of information extraction is the recognition of handwritten numerals. Even though this is an established field, still it becomes difficult to recognize numbers with an efficient optimization. Due to higher levels of complexity and storage, training of such a model using large amount of data often fails. A convolutional neural network (CNN) with amalgamation of mini-batch and stochastic Hessian-free optimization (HybOpt) technique is used in this paper to get predictions that are accurate and converge more quickly. When solving quadratic equations, which intensely depend upon computation and storage, a second-order approximation is applied to escalate speed. The proposed technique is also using a repetitive minimization method for quicker convergence with an arbitrary initialization. The effectiveness of the proposed method is examined by doing study on standard MNIST dataset. The CNN is trained up to 15 layers using HybOpt and results are compared with existing optimization techniques such as MBSGD, SGD, SHF, HFO and NCG. The dataset was divided into 85% training and 15% testing images. Parameters for example accuracy, f1-score, precision and recall are found. Additionally, running time against epochs are also observed. In all the parameters, it was found that the proposed method produces the best results.

Keywords: Accuracy, CNN, Deep learning, Handwritten Digit Recognition, Optimization.

1. Introduction

The arena of image processing has been seeing amazing developments in recent years in high dynamic range imaging. HDR imaging is a technique by which broader range of brightness and intensity levels of an image are captured; therefore, maintaining the details in both bright and dark portions of an image. This technique has applications in all the spheres such as healthcare, agriculture, cinematography, photography to name a few. Deep learning approaches are becoming popular as the need for real time HDR imaging continues to grow as they provide effective and efficient performance.[1]

The identification of handwritten digits is a vital job in the area of image processing. Its applications range from the recognition of characters automatically to the examination of documents. Digit recognition in handwritten images, on the other hand, is a difficult process owing to the fact that a number of fundamental issues are involved.

These issues come from the fact that handwritten characters are difficult to read. The fact that different people have different writing styles is one of the most significant problems that must be overcome in order to recognize handwritten digits. Handwritten numbers may display a significant amount of variability in terms of size, form, slant, and spacing, in contrast to printed text, which follows standard typographic norms. As a consequence of this, differentiating between numerals that seem to be visually identical, such as "6" and "9" or "1" and "7," may be difficult even for human readers [2]. Recognition systems face a difficult challenge as a result of this variability since they are required to cope with an ever-expanding universe of potential variants. Furthermore, people may acquire distinctive writing patterns over the course of time, which further complicates the process of reliably detecting handwritten numerals across a variety of authors and circumstances.

The existence of these uncertainty highlights the need of creating strong feature extraction and classification approaches to handwritten digit recognition.

Although real time HDR imaging has its own different set of issues, mainly when it comes to the detection of digits in handwritten images. This makes correct detection of handwritten pictures a challenging endeavor since handwritten images often display a variety of writing styles, varied ink intensities, and complicated backdrop textures. A basis for effective performance in digit identification is provided by the incorporation of deep learning algorithms into the HDR image. These approaches provide viable solutions to overcome the above issues.

2. Literature survey

Ahmed, Syed et al. (2023) [3] have used EfficientDet-D4 and attempted to solve the limitations that were previously addressed. When the input is first labeled, the region of interest is displayed exactly as intended. Then, these images are used to train the EfficientDet-D4 model to identify and classify the numbers in the proper classes, which range from 0 to 9. With this, they achieved an average accuracy of 99.83% on the MNIST dataset.

Ogundijo, Michael. (2023) [4] found that the performance of the recognition technique is constantly influenced by a variety of factors, such as the writing style, the size of the digits, and the recognition speed. The purpose of this study is to attempt to present a satisfactory answer to the issue by using two separate inquiries. The kind of dataset that is most often used and the reasons why it is the most effective for determining handwritten digit identification, as well as the machine learning technique that offers the highest performance based on accuracy.

P V Siddhartha et al., (2023) [5] discussed an approach to handwritten digit recognition is which is based on the concept of CNN. The network was trained with MNIST dataset along with loss function and were able to get a recognition accuracy of 99.3 %.

Nawaz, Marriam (2023) [6] presented a deep- learning-based model known as Fast-RCNN where they have classified the input dataset into 10 categories, ranging from 0 to 9. For the purpose of evaluating our model, the difficult dataset known as "MINIST" is used. The results of the experiments proved that the method used is capable of precisely classifying numerical images even when there are changes in the writing style, as well as when there is noise, blurring, and other such things present in the input images.

Abdulraheem, Abdulkabir (2023) [7] used generative adversarial networks and enhanced the dataset to improve classifier’s performance. Additionally, they developed the CNN for engraved digit (CNN- ED) model and achieved accuracy of 99.88%.

Sengan, Sudhakar et al., (2023) [8] developed a variation of CNN and applied image processing methods as convolutional layers at the first layer. With the exception of the I/O layers, which are responsible for solving minor matrix equations, all of the neurons that are included inside a single layer typically do simple arithmetic operations. This is the way that a layer is first formed. The training and classification of Machine Learning were selected from the MNIST database, and the results were improved by utilizing an improving algorithm.

Sharma, Mayank et al, (2023) [9] throughout their study, tried to implement a model that is capable of recognizing the numbers and therefore challenges. The algorithm developed was able to identify the numbers that have been provided with an accuracy of 98.40%. For training model, CNN was used.

Hadjadj, Ismail et al., (2023) [10] provided a novel approach that makes use of encryption strategies at the stage of feature extraction. Additionally, this approach is particularly effective for handwritten digit identification since it is less impacted by differences in form and slant.

Zeki, Umut et. Al., (2023) [11] proposed a new handwriting verification method that verifies numbers from the handwriting of children who are enrolled in special education programs. The implementation makes use of a CNN that was developed and trained in bottom-up approach. A recognition accuracy of around 94% is achieved by the system.

Peng, Chao-Chung et al., (2023) [12] developed a handwritten digit recognition system that is both low-cost and requires a minimal amount of computing. A logistic regression classifier that is based on PCA was used. For the purpose of establishing whether or not the newly created method for image recognition is successful, performance assessments are carried out using the widely known MNIST dataset. The suggested approach is able to generate higher prediction results with a lower model size, which is 18.5% better than the classic logistic regression.

Singh, Divya et al., (2022) [13] presented DDRNet that is created on Deep CNN with an aim to limit features and parameters in order to make the model lightweight, whereas simultaneously optimizing accuracy of digit recognition using an adaptive voting (AV) strategy. CNN is responsible for identifying each individual digit, whereas DCN with AV and VWCRF approach are used to identify strings or digits that are doubtful. These techniques were first developed using the YOLO algorithm. DDRNet solution attains an accurateness of 99.4% deprived of error variations.

Agrawal, Vanita et al., (2022) [14] developed a hybrid model on the recognition of handwritten digits. The testing is carried out on the EMNIST and DIDA datasets. Both attained an accuracy of 99.89% and 99.73%, respectively, when it came to the 10-fold cross-validation approach.

Gupta, Akanksha et al., (2021) [15] used different classifiers such as KNN, SVM, and CNN for hand written digit recognition. With the help of a predetermined dataset, these classifiers are trained to convert any digital scan document. First, the image is preprocessed, then segmented and finally the classifier is used to detect the scanned document. This procedure is repeated four times before the final identification step. Training purposes are served by the MNIST dataset and found that CNN maintains a high level of accuracy.

Fateh, Amirreza et al., (2021) [16] proposed a model that is independent of language and is built on a robust CNN. The transfer learning was also included to improve the picture quality. A total of six distinct languages were used in the testing of the proposed system. When it came to identifying other languages and the digits that are connected with them, the findings showed an average accuracy of up to 99.8%.

Yahya, Ali et al., (2021) [17] suggested the addition of an additive white Gaussian noise with a value of $\sigma = 0.5$ to the MNIST dataset. This is done with the intention of simulating the natural effects that may have an impact on the quality of images in the real world and achieved accuracy of 99.98% and 99.40% with 50% noise during recognition.

Savita, Ahlawat et al., (2020) [18] attained equivalent accuracy by using a pure CNN design rather than an ensemble architecture. As a result, a CNN design is presented in order to obtain high accuracy. A combination of learning parameters in the process of creating a CNN was also offered, which achieved a recognition accuracy of 99.87% across an MNIST dataset.

Yellapragada, Bharadwaj et al., (2020) [19] created CNN using MNIST dataset to recognize handwritten digits. The approach that was developed attained an accuracy of 98.51% for the prediction of real-world handwritten digits, with a loss of less than 0.1 percent during training with 60000 digits and 10000 digits during validation process.

Nouri, Houssem Eddine. (2020) [20] offered a case study that consisted of recognizing handwritten numbers from academic transcripts. By designing a 6-layered CNN, accuracy of training and testing are achieved as 98.45% and 98.01 % respectively.

Ali, Saqib et al., (2020) [21] found that the CNN-ELM- DL4J strategy as developed by them accomplishes better results than the traditional CNN models in terms of both accuracy and the amount of time required for calculation.

Mukhoti, Jishnu et al., (2020) [22] made use of modified version of CNN architecture that is known as LeNet. Extensive trials have shown that the best-case classification accuracy for Bangla numerals is 98.2%, and for Hindi numerals, it is 98.8%.

Table 1. Summarized Literature Survey

SN	Author(s) & Year	Problem/Objective	Classifier Used	Dataset/Results
1	Ahmed, Syed et al., 2023	To recognize handwritten digits in spite of issues such as variations in writing styles, visual distortions and noise.	EfficientDet-D4	MNIST: 99.83%, USPS: 99.10%

2	Ogundijo, Michael, 2023	To review and identify the most effective dataset and model for hand written digit recognition.	KNN, Decision Trees, DNN, SVM	MNIST 100%
3	Siddhartha et al., 2023	To improve recognition of handwritten digits.	CNN	MNIST 99.3%
4	Nawaz, Marriam, 2023	To resolve challenges for example variations in writing style and distortions in input images for identifying hand written digits.	Fast-RCNN	MNIST
5	Abdulraheem, Abdulkabir, 2023	To improve accuracy for digit recognition for engraved digits.	CNN-ED	Engraved digit 99.88%
6	Sengan, Sudhakar et al., 2023	To resolve complexity issues in recognizing Arabic numerals.	CNN-based	Arabic numeral dataset
7	Sharma, Mayank et al., 2023	To implement a system to recognize digits for automation.	CNN	98.40%
8	Soni, Sarvesh et al., 2023	To compare CNN with other deep learning networks for hand written digit recognition.	CNN	Various datasets >90%
9	Runwal, Rutuj et al., 2023	To build a portable online application for digit recognition.	Cloud-based DL	MNIST
10	Zeki, Umut et al., 2023	To resolve the difficulty in recognizing handwriting for special education students.	CNN	94%
11	Peng, Chao-Chung et al., 2023	To develop a low-cost system resilient to image rotations.	PCA-based LR	MNIST
12	Singh et al. (2022)	To improve handwritten digit recognition particularly with overlapping digits.	DDRNet using CNN	MNIST 99.4%
13	Agrawal et al. (2022)	To improve handwritten digit recognition by addressing limitations of CNNs.	CNN+ViT and MLP.	EMNIST: 99.89% and DIDA: 99.73%
14	Kamble et al. (2022)	To overcome challenges in handwriting character recognition with high false acceptance rates.	Adaptive threshold-based method	MNIST and VDM
15	Li et al. (2022)	To improve optimization performance of Sand Cat Swarm Optimization (SCSO) algorithm for various engineering problems.	SE-SCSO	Comparative Analysis
16	Gupta et al. (2021)	To enhance handwritten digit recognition system in the context of digital document processing.	KNN, SVM, CNN	MNIST
17	Albahli et al. (2021)	To improve handwritten digit recognition by addressing challenges like variation in handwriting styles, noise and distortions.	Fast-RCNN with DenseNet-41	MNIST

18	Fateh et al. (2021)	To developing a multilingual handwritten numeral recognition system to address challenges.	CNN	Six languages/ avg: 99.8%.
19	Yahya et al. (2021)	To improve CNN classification algorithms by addressing issues like filter size, data preparation, noise and dataset limitations.	CNN	MNIST
20	Savita, Ahlawat et al. (2020)	To investigate CNN-based handwritten digit recognition to achieve high accuracy while reducing complexity as compared to ensemble architectures.	CNN-based	MNIST 99.87%
21	Yellapragada, Bharadwaj (2020)	To recognize digits written in diverse styles.	CNN	MNIST 98.51%
22	Nouri, Houssem Eddine (2020)	To identify handwritten digits in academic transcripts using a CNN model.	CNN	MNIST: 98.45%
23	Ali, Saqib et al. (2020)	To identify the need for a robust model for efficient handwritten digit recognition with improved feature extraction.	CNN-ELM	MNIST and USPS
24	Mukhoti, Jishnu et al. (2020)	To improve the recognition of Bangla and Hindi handwritten digits using CNN.	CNN	Bangla: 98.2%, Hindi: 98.8%

3. Proposed Scheme

Mini-Batch Gradient Descent: It is a vital optimization technique that is used in machine learning as well as deep learning for training models efficiently on large datasets. It is a variant of the gradient descent technique that divides dataset in smaller batches & updates model parameters iteratively based on the average gradient computed from each batch [23]. There are a number of reasons to prove that MBGD is superior to other algorithms of gradient descent:

- As a result of the reduction in the periodic parameter variance update, stable convergence has occurred.
- In order to evaluate the gradient in an effective manner, high matrix optimizations are used.

Below is the algorithm to show the iterations over batch size as 1000.

Algorithm 1 (Iterations over mini batches with size as 1000)

Step 1: Start

Step 2: Repeat step 3 n times. This is denoted by n_epochs.

Step 3: In each iteration, shuffle the dataset by using a randomized function.

Step 4: Divide the dataset into smaller chunks that is mini batches and set the batch size of dataset as 1000 and repeat step 5.

Step 5: For each mini batch, calculate the gradient and update the model's parameters until get the desired results.

Step 6: Stop

Levelling Mini-Batch Gradient Descent (MBGD) Algorithm

Since, the MBGD technique does not satisfy noble convergence, so the second-order derivation is being considered. For multi- dimensional data such as images, the Newton's approach, which is represented by Equation 1, is widely considered to be a good method. For the function, $f\colon Rn \rightarrow R$,

$$x_{n+1} = x_n - (H(f)x)\nabla 1 \quad f(x)$$

According to Sainath et al. (2011) [24] and Lin et al. (2017) [25] , one of the most significant drawbacks of Newton's technique is that it does not provide any assistance at all in the calculation of the Hessian matrix. During the process of calculating the Hessian matrix that is necessary for a number of other techniques, the gradient value is assessed using back-propagation in neural networks. According to Bottou et al. (2016) [26], one of the most significant disadvantages of adopting multi-dimensions is that it requires much more compute and storage space to forecast matrix representation. Hence, the Hessian Free Optimization procedure demands the application of second-order optimization in order to eliminate a variety of consequences on MBGD.

Hessian - Free Optimization: According to Martens et al. (2012) [27] and James et al. (2010) [28], HFO is a more accurate approach to handle both the concerns of HFO as it provides smaller storage matrix area as well as perfect NN solution. This is the reason why the HFO strategy is more efficient. When it comes to the interconnection model that employs the differential objective function, HFO performs exceptionally well. This particular HF realization for RNN and FF offers its applicability to Multi-Dimensional Neural Networks as stated by Simonyan et al. (2015) [29] and Lin et al. (2018) [30]. Let us minimize the function f as stated by $f: Rn \rightarrow R$. Algorithm for HFO is shown below:

Algorithm for Hessian Free Optimization

Step 1: Start

Step 2: Assign i=0 and xi = x0 be initial guess.

Step 3: Compute the gradient and Hessian matrix at the current xn.

Step 4: Repeat steps 5 and 6 until xn have converged:

Step 5: Consider the following Taylor expression of 'f' as:

$$f(x+\Delta x) \approx f(x) + \nabla f(x)T\Delta x + \Delta x\, TH(f)\, \Delta x$$

Step 6: Compute xn+1 using Conjugate Gradient algorithm for quadratic functions on the existing Taylor expansion.

Step 7: Stop

In the first stage, the values of $'x'$ are selected at random, and in the second step, the evaluation of $(f)(xn)$ is constructed via back-propagation. The conjugate gradient for the 'i' iterations is calculated in step 6 and there is no stopping factors involved. As a result, the iterations are prolonged using certain heuristic principles in order to extend the iterations and provide more accurate predictions of superior convergence. Because of this, the Hessian properties ought to be satisfied by the second-order information obtained via NN. The second-order approaches are used for the heuristic embracing of error surfaces associated to superior convergence in Hessian, which is an expanded form of Newton's method (Wiesler et al. 2013) [31]. Equation 2 is stating the same numerically as follows:

Figure 1 : Hessian matrix of a neural network from (Source: Chapelle et al. 2011) [32]

$$H(e) = \begin{vmatrix} \frac{\partial^2 e}{\partial w_1^2} & \frac{\partial^2 e}{\partial w_1 \partial w_2} & & \frac{\partial^2 e}{\partial w_1 \partial w_n} \\ \frac{\partial^2 e}{\partial w_2 \partial w_1} & \frac{\partial^2 e}{\partial w_2^2} & \cdots & \frac{\partial^2 e}{\partial w_2 \partial w_n} \\ \frac{\partial^2 e}{\partial w_n \partial w_1} & \frac{\partial^2 e}{\partial w_n \partial w_2} & & \frac{\partial^2 e}{\partial w_1^2} \end{vmatrix} \qquad \begin{aligned} (HV)_i &= \sum_i^n \frac{\partial e}{\partial w_2 \partial w_j} V_j \\ (HV)_i &= \nabla \frac{\partial e}{\partial w_i} V \end{aligned}$$

The letter 'i' and 'j' specifies row and column indices respectively. The directional derivative of 'i' through 'j' towards 'i' is provided in Equation 3. To optimize the aforementioned, use the following Equation 4 and use the concept of finite differences:

$$HV = \frac{\nabla f(e+\epsilon v) - \nabla f(e)}{\epsilon}$$

Using mathematical vector optimization, the Hessian calculation is removed by concentrating on the Hessian approximation, which results in a significant reduction in both the capacity for computing and storage.

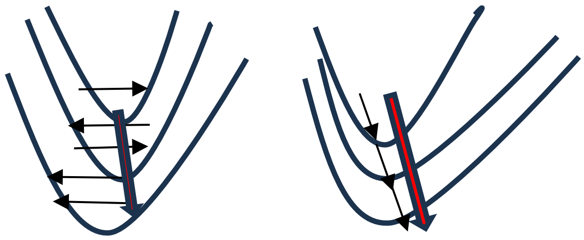


Figure 2: (a) SGD (b) Hessian Free Optimization

Proposed Technique (HybOpt)

The HybOpt algorithm is a hybrid form of the mini-batch (MB) and stochastic hessian-free optimization (SHFO) methods. MBGD is developed from a gradient descent method. This method divides the training dataset into a smaller number of batches. These batches are then used to compute the error as generated by the model and update its coefficients. In order to limit the variance, the average and cumulative values of all batches are carried out and analyzed. The mini-batch gradient descent algorithm is able to successfully handle efficiency of batch gradient & resilience of stochastic gradient descent strategies. The flow chart of proposed work is shown below:

Figure 3: Workflow of generic hand written digit recognition system

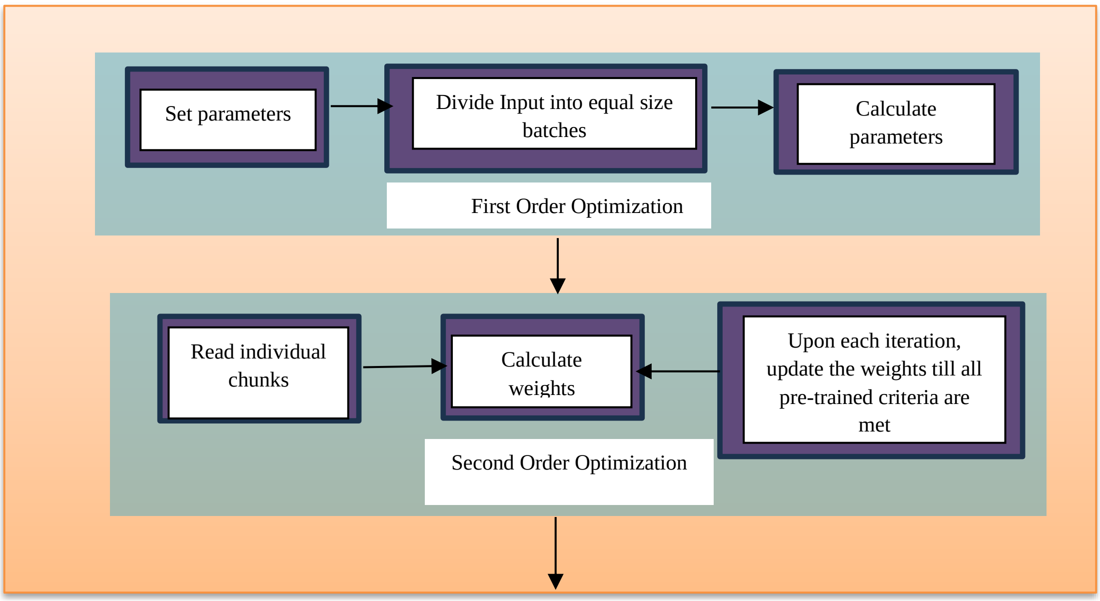
Input Image: Hand written digit images from standard MNIST dataset are taken to feed into the model. The identification procedure by model was done by a line-by-line scan of the input images.

Pre-Processing: At this stage, images were resized. Images preprocessing can be done by considering criteria such as height, width and number of RGB channels. After this, noise was reduced by using median filtering technique and images were smoothened. The resultant smoothened images were applied to greyscale image then.

Classifier: After preprocessing, the grey-scale images are put into CNN classifier, where its weights are updated on each iteration.

Proposed HybOpt Optimizer: During training phase, smaller batch size (in our case 1000) is chosen to train the model so as to get fast convergence. Once batch size is fixed, it can not be altered by any further parameters. Additionally, it is necessary to observe the training and validation curves during computation in relation to the overall amount of time required for optimization during each iteration. The below figure shows the working of proposed HybOpt Optimizer in pictorial form.

Figure **4.**



Proposed HybOpt Optimizer

Below algorithm shows the working of HybOpt Optimization:

Step 1: Start

Step 2: Divide the image data set into n mini batches, say B1, B2, ..., Bn.

Step 3: Repeat step 3.1 to step 3.4 until F number of iterations (till convergence).

3.1: Choose a random mini batch Bt from the set {B1, B2, ..., Bn}.

3.2: Divide the selected mini batch Bt into m sub-partitions, say Bt1, Bt2, ..., Btm

3.3: Repeat step 3.3.1 to step 3.3.2, for each sub-partition Btj (where j = 1, 2,., d) in parallel.

3.3.1: Calculate the gradient descent for the sub- problem.

3.3.2: Update the weight for sub-partition.

3.4 Calculate the average of all the weights obtained from all the sub-partitions to get the final weight.

Step 4: Display the final weight vector as result.

Step 5: Stop.

Training and Evaluation: The attained weights are then converged and averaged towards a specific value, thus recovering the edge values from HD images. In this phase, the proposed model is trained using MNIST dataset and the images are validated with time factor and batch size.

5. Experimentation Environment:

In the study, we selected MNIST dataset with number of images as 70,000 out of which approximately 85% images that is 60,000 are used as training images and approximately 15% images that is 10000 are used as testing images. Each image is of 28*28 pixels size. We then compared our study with the results obtained from the existed models. Below is the figure representing sample image from MNIST dataset.

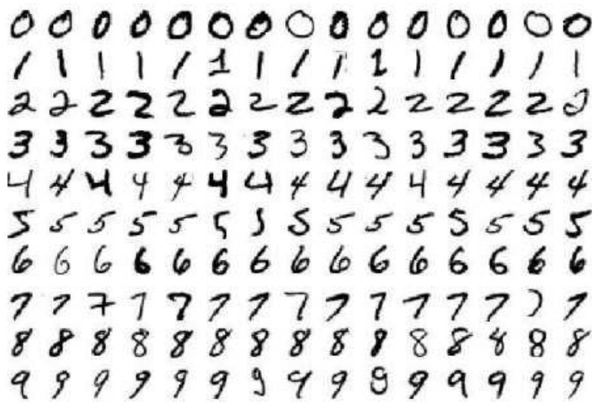


Figure 5. Sample of MNIST dataset

The following hyper-parameters in the proposed model are chosen:

1. Randomly choose the mini batch of size 1000.
2. Iteration ends on reaching the count 1000 because if iterations will be more than cost of computation and convergence rate will also be more.

During experimentation, input image from our chosen dataset in greyscale form is applied to the model. The proposed model and existing models were realized and results were studied and compared. After analyzing the results, we found that a general optimizer without being pre-trained can effectively and efficiently learn various DNN techniques. We further found that the proposed model was very effective in resolving the under fitting problem as it produces training error of 0.90. Since we have used stopping constraint earlier thus reducing the training error to 65.7% in compared to HFO which produces training error of 1.40 without any stopping condition. Our proposed hybrid method optimizes in much faster way. The study is also compared with some existing optimization techniques.

Table 2. Comparison with other existing Optimization Models

Name	Acronym used	Learning Rate	Batch Size
Stochastic Gradient Descent	SGD	Fixed (0.02)	Small (20)
Non-linear Conjugate Gradient	NCG	Dynamic	Large (in thousands)
Hessian-Free optimization	HFO		
Stochastic Hessian-Free optimization	SHFO		
Mini Batch Stochastic Gradient Descent	MBSGD	Fixed (0.02)	
Proposed Model	HybOpt	Fixed (0.02)	

This paper focuses on the suggested HybOpt optimization over SGD, NCG, HFO, SHFO and MBSGD without being pre-trained, evaluations are carried out on the parameters such as training iteration, epochs, parameters, time and convergence rate. It was found that each model has its different average amount of time to complete their execution. The same is summarized in the below table:

Table 3. Comparison of average running time

Name	Acronym used	Average Time to complete (in seconds)
Stochastic Gradient Descent	SGD	7
Non-linear Conjugate Gradient	NCG	3.5
Hessian-Free optimization	HFO	4.5
Stochastic Hessian-Free optimization	SHFO	8
Mini Batch Stochastic Gradient Descent	MBSGD	4.2
Proposed Model	HybOpt	3.2

Additionally, comparisons based on parameters are shown below in the form of graphs and charts. Figure 6 shows the comparative analysis of the result of accuracy found when training images (as per the number specified) fed into the different models. At the size of 50,000, it is clearly shown from the graph that proposed model gives the best accuracy as 0.74 while others give as less as 0.35 and as more as 0.61.

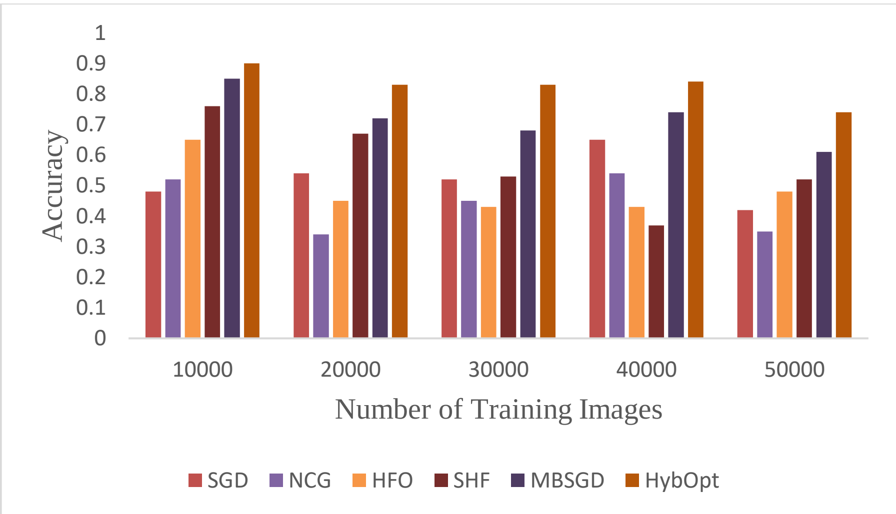


Figure 6: Comparative Result of Accuracy v/s No. of Training Images

Figure 7 shows the comparative analysis of the result of accuracy found when we tested the images (as per the number specified) on the different models. At the minimum size of 2000, the proposed model gives the best accuracy as 0.80 while others give in the range of 0.42 to 0.62 and at the maximum size of 10,000, the proposed model gives the best accuracy as 0.60 while others give as less as 0.32 and as more as 0.56.

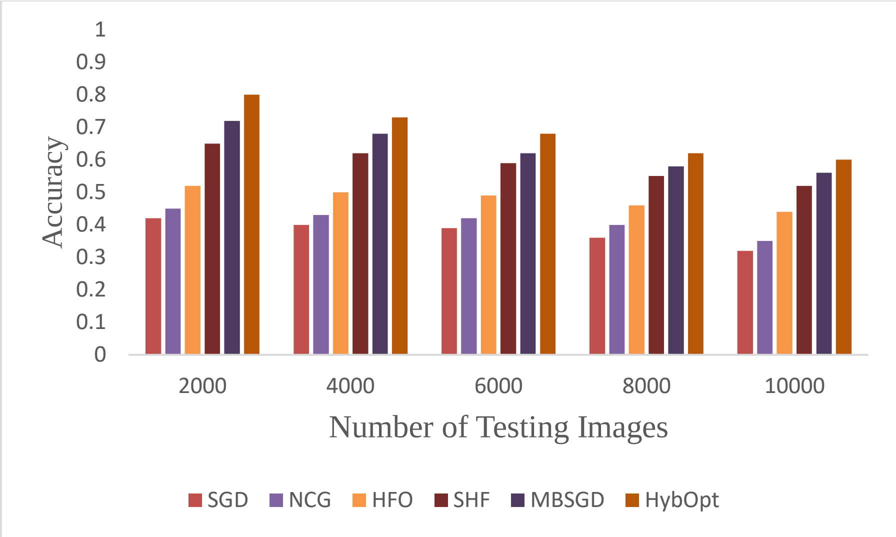


Figure 7: Comparative Result of Accuracy v/s No. of Testing Images

Additionally, the study has also compared the running time of various optimization techniques with the proposed one on 350 number of epochs and it was found that the proposed technique has provided the solution in least time of 726 seconds, while others execute ranges from 773 seconds to 875 seconds on the same number of epochs. Hence it can be concluded that proposed technique gives enhanced results than the others being currently in use. Following table shows the same:

Table 4. Comparative Analysis of Convergence time and efficiency

Name	**Acronym used**	**Running time**	**Convergence Efficiency**
Stochastic Gradient Descent	SGD	875 Sec	$O(T)$
Non-linear Conjugate Gradient	NCG	810 Sec	$O(T)$
Hessian-Free optimization	HFO	792 Sec	$O(1/T)$
Stochastic Hessian-Free optimization	SHFO	773 Sec	$O(1/T)$
Mini Batch Stochastic Gradient Descent	MBSGD	788 Sec	$O(1/\sqrt{b}T)$
Proposed Model	HybOpt	726 Sec	$O(1/lnT)$

6. Conclusion

A pre-trained model for the stochastic hessian-free optimization strategy is examined during this study. The model is combined through a mini-batch size of one thousand. A usual process is followed to perform the experiments in order to optimize the error rate of the DNN. With a better convergence rate of $O\ (1/ln\ T)$, proposed model is proved to be better. It was also found that if the size of mini-batch was altered from 1000, its convergence rate does not affect. Hence, the proposed technique was used to carry out handwritten digit recognition. This approach minimizes the amount of memory that is required as well as the amount of time that is spent on calculation. In future, the proposed model will be applied to healthcare and agriculture sector for accuracy and prediction.

References

[1] Yamming, G 2016, ‘Deep learning for visual understanding: A review’, Elesvier- Neurocomputing, vol. 187, pp. 27-48.

[2] Alzubaidi, L, Zhang, J & Humaidi, AJ 2021, ‘Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions’, Journal of Big Data, vol. 8, no. 53.

[3] Ahmed, Syed & Mehmood, Zahid & Awan, Imran & Yousaf, Rehan. (2023). A Novel Technique for Handwritten Digit Recognition Using Deep Learning.Journal of Sensors. 2023. 10.1155/2023/2753941.

[4] Ogundijo, Michael. (2023). Handwritten Digit Recognition Using Machine Learning Models : A Review. 10.13140/RG.2.2.12565.29925.

[5] P. V. Siddhartha et al, "Digit Recognition of MNIST Handwritten Using Convolutional Neural Networks (CNN)," 2023 International Conference on Intelligent Systems for Communication, IoT and Security (ICISCoIS), Coimbatore, India, 2023, pp. 328-332, doi: 10.1109/ICISCoIS56541.2023.10100602.

[6] Nawaz, Marriam & Javed, Ali & Nazir, Tahira & Masood, Momina & Mehmood, Awais & Saleem, Muhammad. (2023). Classification of Hand-Written Digits using Deep-Learning Technique.

[7] Abdulraheem, Abdulkabir, and Im Y. Jung. 2023. "Effective Digital Technology Enabling Automatic Recognition of Special-Type Marking of Expiry Dates" Sustainability 15, no. 17: 12915. https://doi.org/10.3390/su151712915

[8] Deol Gosu, J.S., Subramaniam, B., Nachimuthu, S., Shivasankaran, K., Subburaj, A., Sengan, S. (2023). Comparative Analysis of Handwritten Digit Recognition Investigation Using Deep Learning Model. In: Agarwal, P., Khanna, K., Elngar, A.A., Obaid, A.J., Polkowski, Z. (eds) Artificial Intelligence for Smart Healthcare. EAI/Springer Innovations in Communication and Computing. Springer, Cham. https://doi.org/10.1007/978-3-031-23602-0_4

[9] Sharma, Mayank & Sindal, Pradhyuman & Baskar, M.. (2023). Handwritten Digit Recognition Using Machine Learning. 10.1007/978-981-19-7615-5_3.

[10] Hadjadj, Ismail & Gattal, Abdeljalil. (2023). An Image Encryption-Based Method for Handwritten Digit Recognition. 10.1007/978-3-031-25344-7_3.

[11] Zeki, Umut & Karanfiller, Tolgay & Yurtkan, Kamil. (2023). Person-Dependent Handwriting Verification for Special Education Using Deep Learning. Intelligent Automation & Soft Computing. 36. 1121-1135. 10.32604/iasc.2023.032554.

[12] Peng, Chao-Chung & Huang, Chao-Yang & Chen, Yi-Ho. (2023). Principal Component Analysis-Based Logistic Regression for Rotated Handwritten Digit Recognition in Consumer Devices. Electronics. 12. 3809. 10.3390/electronics12183809.

[13] Singh, Divya & Bano, Shahana & Samanta, Debarata & Mekala, M. S. & Islam, SK. (2022). Deep Learning Inspired Nonlinear Classification Methodology for Handwritten Digits Recognition Using DSR Encoder. Arabian Journal for Science and Engineering. 48. 1-13. 10.1007/s13369-022-06865-8.

[14] Agrawal, Vanita & Jagtap, Jayant. (2022). Convolutional Vision Transformer for Handwritten Digit Recognition. 10.21203/rs.3.rs-1984839/v1.

[15] Gupta, Akanksha & Narwaria, Ravindra Pratap & Solanki, Madhav. (2021). Review on Deep Learning Handwritten Digit Recognition using Convolutional Neural Network. International Journal of Recent Technology and Engineering. 9. 245-247. 10.35940/ijrte.E5287.019521.

[16] Fateh, Amirreza & Fateh, Mansoor & Abolghasemi, Vahid. (2021). Multilingual Handwritten Numeral Recognition Using a Robust Deep Network Joint with Transfer Learning. Information Sciences. 581. 10.1016/j.ins.2021.09.051.

[17] Yahya, Ali & Tan, Jieqing & Hu, Min. (2021). A Novel Handwritten Digit Classification System Based on Convolutional Neural Network Approach. Sensors. 21. 6273. 10.3390/s21186273.

[18] Savita, Ahlawat & Choudhary, Amit & Nayyar, Anand & Singh, Saurabh & Yoon, Byungun. (2020). Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN). Sensors. 20. 3344. 10.3390/s20123344.

[19] Yellapragada, Bharadwaj & Kolla, Bhanu. (2020). Effective Handwritten Digit Recognition using Deep Convolution Neural Network. International Journal of Advanced Trends in Computer Science and Engineering. 9. 10.30534/ijatcse/2020/66922020.

[20] Nouri, Houssem Eddine. (2020). Handwritten Digit Recognition by Deep Learning for Automatic Entering of Academic Transcripts. 10.1007/978-3-030- 63319-6_53.

[21] Ali, Saqib & Li, Jianqiang & Pei, Yan & Aslam, Muhammad & Shaukat, Zeeshan & Azeem, Muhammad. (2020). An Effective and Improved CNN-ELM Classifier for Handwritten Digits Recognition and Classification. Symmetry. 12. 15. 10.3390/sym12101742.
[22] Mukhoti, Jishnu & Dutta, Sukanya & Sarkar, Ram. (2020). Handwritten Digit Classification in Bangla and Hindi Using Deep Learning. Applied Artificial Intelligence. 34. 10.1080/08839514.2020.1804228.
[23] Li, Mu & Zhang, Tong & Chen, Yuqiang & Smola, Alexander. (2014). Efficient mini-batch training for stochastic optimization. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 10.1145/2623330.2623612.
[24] Sainath et al 2011, 'Making deep belief networks effective for large vocabulary continuous speech recognition', Automatic Speech Recognition and Understanding (ASRU), IEEE Workshop on IEEE, pp. 30-35.
[25] Lin, Dollar, P, Girshick, R, He, K, Hariharan, B & Belongie, S 2017, 'Feature pyramid networks for object detection', Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2117-2125.
[26] Bottou, FE, Curtis & Nocedal J 2016, 'Optimization methods for large-scale machine learning, PHI, USA.
[27] Martens & Sutskever, I 2012, 'Training deep and recurrent networks with hessian-free optimization in neural networks', Springer, vol. 2, pp. 479-535.
[28] James Martens, Leon Bottou & Michael Littman 2010, 'Deep learning via hessian-free optimization', Proceedings of the Twenty-Seventh International Conference on Machine Learning, vol. 10, pp. 735-742.
[29] Simonyan & Zisserman, A 2015, 'Very deep convolutional networks for large- scale image recognition', published as a conference paper at ICLR, pp. 1-12.
[30] Lin, F, Lin, Y, L, F, Cai & D, Cao 2018, 'Chinese character captcha recognition and performance estimation via deep neural network', Neuro computing, vol. 288, pp. 11-19.
[31] Wiesler, S, Li, J & Xue, J 2013, 'Investigations on hessian free optimization for cross-entropy training of deep neural network', in Inter Speech, pp. 3317-3321.
[32] Chapelle & Erhan, D 2011, 'Improved preconditioned for hessian free optimization', in NIPS Workshop on Deep Learning and Unsupervised Feature Learning.