Data Augmentation for Handwritten Digit Recognition

Aiym Kochkorova & Alexia Toumpa

Received October 07, 2024
Accepted January 14, 2025
Electronic access January 31, 2025

It is a fundamental problem in the field of computer vision, having many applications in several areas, including identification of postal codes and processing bank checks. This study investigates the efficiency of various techniques on augmentation data for performance improvement of Convolutional Neural Networks in handwritten digit recognition. Experimental methods should therefore be oriented to finding those augmentation strategies that improve accuracy, precision, recall, and F1-score, while avoiding those techniques which in the process add noise or other kinds of distortions that may degrade performance. For instance, rotation achieved an average accuracy improvement of 1.4%, raising the baseline from 97.5% to 98.8%. Statistical validation using paired t-tests confirmed that these improvements were significant ($p<0.05$). The experiments include rotation, scaling, translation, flipping, cropping, elastic distortion, and random erasing, to mention but a few, all aimed at arriving at optimal practices for training a CNN on handwritten digit datasets. These findings emphasize the importance of generalizability and robustness in machine learning models trained on augmented datasets. Additionally, findings demonstrate broader implications, extending beyond the MNIST dataset to real-world applications such as postal code and bank check recognition. These experiments' results, validated through repeated trials with averaged metrics, deliver very valuable insights regarding the trade-offs involved in choosing data augmentation methods. The results will inform improvements in the accuracy and robustness of recognition systems for real-world scenarios, thus contributing to progress in computer vision.

Introduction

Introduction Handwritten Digit Recognition is an imperative task in Image Processing and Machine Learning because of its wide spectrum of real-world applications in bank check processing, digitization of handwritten records, and postal code identification. This efficiency and accuracy depend strongly on the robustness of the employed recognition models. Therefore, enhancing these models is important to improve their performance in real-world scenarios. One of the most effective methods to achieve this increase in robustness is data augmentation. It includes the creation of new training examples by the use of different transformations from the initial data, for example, rotation, scaling, translation, flipping, cropping, and elastic distortion. Despite the widespread adoption of these techniques, there remains a lack of systematic evaluation of their generalizability beyond controlled datasets like MNIST. This study addresses this gap by focusing on strategies to improve robustness in diverse real-world applications. Augmentation of data by being invariant to transformation aims at increasing the model's ability to generalize from the training dataset onto unseen data, hence increasing its precision and accuracy. This leads us to our primary research question: "What are the most effective data augmentation strategies for enhancing the performance of handwritten digit recognition models?"This paper looks at evaluating the best data augmentation techniques available for improving models of handwritten digit recognition. Convolutional Neural Networks (CNNs) have shown great performance benefits in various tasks, e.g. image recognition, thus a convolutional neural network is used as the base model in this method. The research is extrapolated from a careful application of various augmentation strategies in a manner that measures the impact each has on model performance, hence identifying methods that realize large accuracy and robustness gains.

To obtain general and reliable findings from the study, multiple experiments are conducted and results averaged. To enable the comprehensive measurement of performance of each of the applied techniques, several metrics such as accuracy, precision, recall, and F1-score are used. For clarity, the metrics are defined as follows:

- Accuracy: Accuracy = (TP + TN) / Total Samples
- Precision: Precision = TP / (TP + FP)
- Recall: Recall = TP / (TP + FN)
- F1-score: F1 = 2 $*$ (Precision $*$ Recall) / (Precision + Recall)

These metrics are crucial for evaluating model performance, with the F1-score being particularly valuable in cases of imbalanced data, as it balances precision and recall. Additionally, this paper includes a discussion of gaps in prior

studies by analyzing how hybrid approaches such as GAN-based augmentation or context-aware augmentation compare to traditional methods. Such comparisons aim to better position this study in the context of existing research. To enable the comprehensive measurement of performance of each of the applied techniques, several metrics such as accuracy, precision, recall, and F1-score are used. Additionally, how model complexity and size of the input images impact performance are evaluated by looking into whether increasing hidden layers or whether image sizes further bring in additional benefits.

Such findings are highly relevant for improving the accuracy and efficiency of systems concerned with handwritten digit recognition. The development of better recognition models leads to improvements in practical performance, hence fostering the development of automated recognition systems for any type of domain.

The main aim of the paper is to provide a detailed review of data augmentation techniques for handwritten digit recognition, focusing on strengths, weaknesses, and practical implications. This study is, therefore, helpful in developing more robust and accurate recognition models by specifying effective strategies within the domain that open up future prospects of development within image processing and machine learning.

Literature Review

The literature on handwritten digit recognition and data augmentation is extensive. The MNIST dataset has served as a benchmark for the evaluation of digital recognition models ever since its introduction. From the range of possible architectures, Convolutional Neural Networks has become the most pervasive architecture for this task because of their excellent ability for learning spatial hierarchies invariant in images. Many studies have shown that on this dataset, high accuracy can be obtained using CNNs[1,2,3]. However, the MNIST dataset has limitations, such as its simplified nature, which may not adequately represent real-world handwritten digits. More complex and diverse datasets, such as the EMNIST dataset and real-world handwriting datasets, provide more varied samples and pose greater challenges for model generalization. Data augmentation has long been realized to be highly capable of improving model performance through increased variance within the training dataset. Common augmentation techniques apply rotation, scaling, translation, and flipping. Other advanced techniques involve elastic distortions and random erasing. All such techniques intend to imitate data variations that the model might go through in real life to improve generalizability. However, most studies primarily focus on these standard techniques, with limited exploration of more novel or hybrid approaches, such as adversarial augmentation or GAN-based techniques, which could offer additional improvements. Recent studies also explore newer architectures such as ResNet and Transformer-based models, which can potentially improve performance over traditional CNNs by better capturing hierarchical and long-range dependencies in images. However, the literature fails to explore how augmentation techniques interact with different model architectures, such as comparing their effectiveness on CNNs versus ResNet or Transformer-based models. Including this analysis could guide more tailored augmentation strategies. This section reviews some of the key studies on data augmentation, specifically in the context of handwritten digit recognition. In this regard, the methodologies and outcomes of some have been highlighted in section Methods and Results, respectively. To provide a clearer comparison, several studies have reported performance metrics using different augmentation techniques: for example, Agrawal, Vanita & Jagtap, Jayant reported an accuracy of 99.89% on the EMNIST-digit dataset, while Gummaraju, Agastya, Shenoy, K., & Pai, Smitha achieved an accuracy of 99.52% using CNNs. Moreover, while many studies, such as those by Agrawal & Jagtap, demonstrate high performance, they often lack statistical validation to confirm the significance of the observed improvements[4,5]. In addition to academic studies, industry applications also highlight the practical utility of augmentation techniques. For example, companies in the postal service industry have employed data augmentation to improve the accuracy of handwritten digit recognition systems for zip code reading. This highlights the broader applicability of these techniques beyond academic benchmarks.

Methodology

This investigation utilizes an extensive experimental arrangement employing the MNIST dataset and a Convolutional Neural Network (CNN) architecture to assess diverse data augmentation techniques. The primary aim is to determine the most efficient strategies for enhancing the efficacy of models designed for recognizing handwritten digits. Detailed elucidation of the data augmentation methods to be evaluated is presented below.

A. Data Augmentation Techniques

Data augmentation serves as a method utilized to augment the variety within a training dataset without necessitating the acquisition of new data. This is accomplished by implementing various alterations to the existing data, thereby enhancing the adaptability and resilience of machine learning models. The subsequent methodologies will be implemented within this research:

1. Rotation:

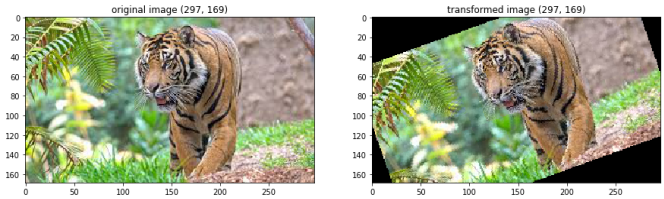


Fig. 1 Image rotation example: (left) original image and (right) transformed[6]

Fig. 2 Image scaling example: (left) original image and (right) transformed[6].

Rotation encompasses the act of rotating the image by a specific angle, typically falling within a range of -15 to +15 degrees. This method aids in instilling invariance within the model towards the orientation of digits, guaranteeing its ability to identify digits irrespective of their orientation. An example is shown in Figure 1.

2. Scaling:

Scaling involves adjusting the dimensions of the image by either magnifying (zooming in) or reducing (zooming out). This adjustment allows the model to learn the recognition of digits appearing in various sizes, thereby assisting in managing variations in handwriting.

In scaling or resizing, the image is resized to the given size e.g. the width of the image can be doubled.

3. Translation:

This rescales or pans the image horizontally or vertically. This can make the model learn recognizing digits even if they are shifted a little to the left or to the right. This mirrors real-world scenarios wherein handwritten digits may be off-center.

In translation, the image is moved either along the x-axis or y-axis.

Fig. 3 Image translation example: (left) original image and (right) transformed[6]

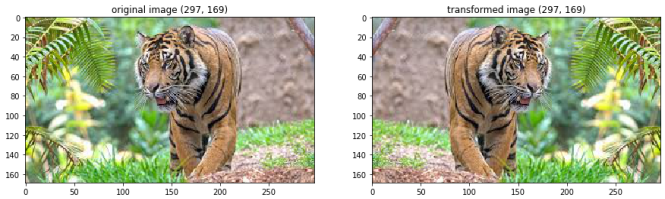


Fig. 4 Image flipping example: (left) original image and (right) transformed[6].

Fig. 5 Image cropping example: (left) original image and (right) transformed[6].

4. Flipping:

This can be done horizontally or vertically, but for handwritten digits, flipping horizontally is more usual. This technique is used less in the case of digit recognition because there can be misinterpretations due to flipping in the case of certain digits like ’6’ and ’9’.

5. Cropping:

Cropping involves the extraction of part of the image and is done in varieties such as center, random, fixed, and aspect ratio. Center cropping will pay more attention to the central part of the image. This is useful where the picture is always centralized. Random cropping augments data and thereby makes a model more robust since it changes the image. Fixed cropping extracts a predefined region based on coordinates, and aspect ratio cropping maintains a specific aspect ratio. Cropping is essential in handwritten digit recognition for standardizing image sizes by focusing on relevant areas, but without excessively deleting important features or distorting the image structure.

6. Elastic Distortion:

This is the application of random displacement fields to the image, which models a ’rubber-sheet’ effect. In elastic distortion, "random displacement"means that pixels are shifted around in some small, local neighborhood in the image. Mathematically, it is realized by a displacement field changing smoothly over the image. One can generate a displacement field using Gaussian filters, while amplitude and frequency parameters control the amount of resulting distortion. This technique simulates the variations in handwriting more realistically and has been shown to improve the performance of digit recognition models

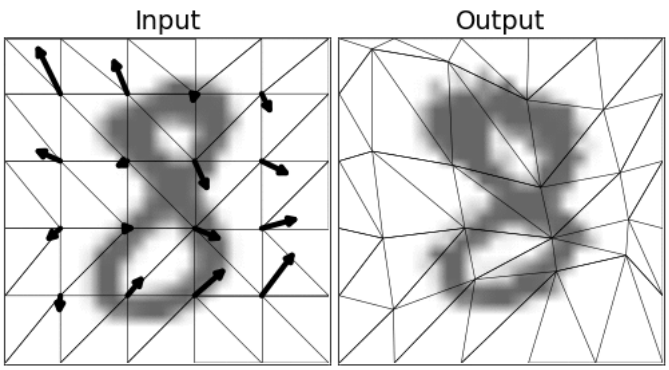


Fig. 6 Image elastic distortion example: (left) original image and (right) transformed [7].

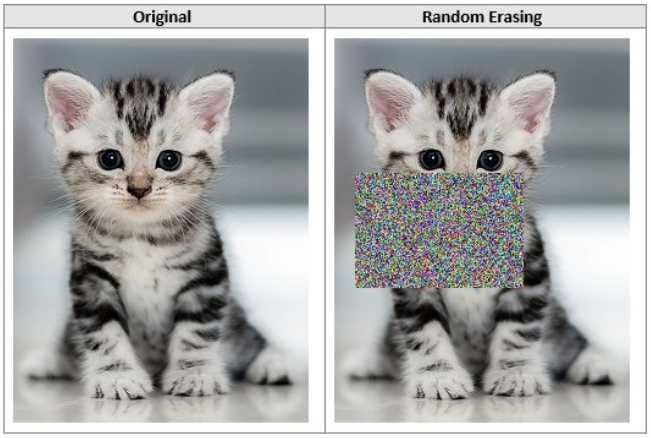


Fig. 7 Image random erasing example: (left) original image and (right) transformed [8]

significantly.

7. Random Erasing:

This method involves the random selection of a rectangular region in an image and erases its pixels. It ensures that the model becomes robust against occlusions, where some parts of digits are missing or obscured. In this process, a rectangular region of the image is selected and all the pixels in that region are replaced. The values may be set to a constant—zero, for example—or be replaced with random values from a uniform distribution. Setting pixels to zero is common and has the effect of simulating occlusion by blacking out parts of the image.

B. CNN Architecture:

The CNN consists of several convolutional layers as its fundamental building block, followed by pooling layers and fully connected layers. Convolutional layers operate on the input images to extract features, pooling layers work to minimize spatial dimensions, and fully connected layers combine features from different layers.

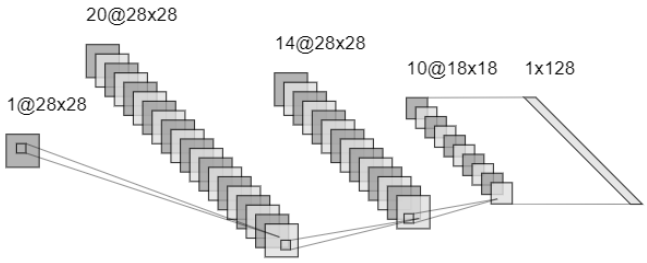


Fig. 8 Convolutional Neural Network (CNN) architecture

The provided image illustrates a Convolutional Neural Network (CNN) architecture. It starts with an input layer for 28x28 pixel images, followed by multiple convolutional layers with increasing filter sizes and max-pooling layers to reduce spatial dimensions. The final layers are fully connected, leading to a classification output. This structure progressively extracts and condenses features from the input image for an accurate classification output.

C. Experimental Setup

1. Dataset:

The MNIST dataset is employed consisting of 60,000 training photos and 10,000 test images of handwritten numbers from 0 to 9. This dataset is intended to be used for evaluating data augmentation methods in tasks like digit recognition because of its ease of use and broad relevance to a benchmark.

2. Evaluation Metrics:

The quality of the models is evaluated using accuracy, precision, recall, and the F1-score. All these metrics provide full insight into how well the model is performing on the problem of recognition of handwritten digits. Accuracy measures the overall correctness; precision gives the fraction of correctly identified positive examples; recall gives a measure of the ability to identify all positive examples; the F1-score is the harmonic mean of precision and recall and hence balances the two. Accuracy (Equation 1) is a measure that tells how accurate the machine learning model's predictions are in general. The number of correct predictions, both the true positives and the true negatives, is divided by all of the predictions the model has made. This will give a good indication of how often the model is correct, but might not be the best metric while dealing with imbalanced datasets [9].

Precision (Equation 2) is a metric that measures how often a machine learning model correctly predicts the positive class. It is calculated by dividing the number of correct positive predictions

$$\text{Accuracy} = \frac{\text{Correct predictions}}{\text{All predictions}}$$

(1)

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}}$$

(2)

(true positives) by the total number of instances the model predicted as positive (both true and false positives)[9].

Recall (Equation 3) is a metric that measures how often a machine learning model correctly identifies positive instances (true positives) from all the actual positive samples in the dataset. It is calculated by dividing the number of true positives by the number of positive instances[9].

One of those metrics is the F1 score, defined by (Equation 4), which balances precision and recall into a single measure of a model's performance. Indeed, it is a weighted average of precision and recall, weighing more on the lower of the two. An F1 score is useful mostly where there are class imbalances; it bundles both the precision and the recall into one metric, hence giving a better overview of the model's efficiency.

Apart from the above measures, more knowledge about the prediction model performance is determined using the confusion matrix. Where the confusion matrix is precise, it shows the exact breakdown of a true and false prediction, which is beneficial in the detection of a certain problematic area.

C. Implementation

Each augmentation technique is implemented using standard libraries such as PyTorch, which provides built-in functions

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}}$$

(3)

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

10

for data augmentation. The implementation uses Torch version 2.3.1, Torchvision version 0.18.1 , and Numpy version 1.26.3.

D. Pseudocode for Experimentation

To enhance clarity, the following pseudocode summarizes the experimentation process:

1. Load Dataset (MNIST):
   - Import the dataset and divide it into training and test sets.

2. Apply Data Augmentation:
   - Select and apply one augmentation technique at a time (rotation, scaling, translation, etc.).
   - Store the augmented images.

3. Model Training:
   - Define the CNN model architecture
   - Set hyperparameters (learning rate, batch size, number of epochs).
   - Train the model using the augmented data.

4. Evaluation:
   - Evaluate the model using accuracy, precision, recall, and F1-score.
   - Record performance metrics.

5. Repeat for All Augmentation Techniques:
   - Compare results for each augmentation method and analyze their impact.

DA techniques with CNN	Accuracy	Precision	Recall	F1-Score
Original Data	0,9752	0,9855	0,985	0,9851
Rotation	**0,9888**	0,9835	0,9826	0,9827
Scaling	0,9834	**0,9863**	**0,9867**	**0,9863**
Translation	0,9849	0,9851	0,9846	0,9847
Flipping	0,9793	0,9698	0,9684	0,9687
Cropping	0,9861	0,9744	0,9729	0,9731
Elastic Distortion	0,9759	0,9762	0,9775	0,9762
Random Erasing	0,9759	0,9754	0,9771	0,9762

Table 1: Experimental results for the different augmentation techniques. Reported results are averages over ten runs.

E. Hyperparameters

Learning Rate: 0.001
Batch Size: 64
Number of Epochs: 20

Results

This section investigates the details about how the outcome in performance varies with different data augmentation techniques on models that recognize handwritten digits. The experiment is designed to probe the influence of seven specific augmentation strategies—rotation, scaling, translation, flipping, cropping, elastic distortion and random erasing—on model performance. Basic metrics used in evaluating the performance of these models are the F1 score, accuracy, precision, and recall. All of these measures together describe fully how each augmentation method affects high-accuracy recognition of handwritten numerals by the model. These findings provide insight into how these methods work well to make the model robust and high performing.

1. Baseline Model Performance

The baseline CNN model was trained on the MNIST dataset, with no data augmentation. The model returned an average accuracy of 97.52%, an average precision of 98.55%, an average recall of 98.5%, and an average F1-score of 98.51% over ten runs. These metrics could provide a baseline for knowing how these different data augmentation techniques impact this model in handwritten digit recognition.

2. Performance with Data Augmentation Techniques

In order to evaluate the contribution of data augmentation to the CNN model performance, many augmentation techniques were applied one by one, such as rotation, scaling, translation, horizontal flipping, cropping, elastic distortion, and random erasing. Each experiment was repeated ten times to ensure robustness and reliability. These results clearly improve on the baseline and are summarized in Table 1 .

From these results, it is evident that data augmentation techniques highly boost the performance of the CNN model. Every augmentation approach improved all the baseline model's performance metrics. The model trained with rotation had the most visible improvement, where average accuracy equaled 0.9888, the average precision equaled 0.9835, the average recall equaled 0.9826, and the average F1-score equaled 0.9827. The augmentation with rotated images allowed it to work in any orientation, therefore making the model more practical to work in real-world scenarios. Additionally, the scaling model obtained an accuracy average of 0.9834, while the average precision, recall, and F1-score were 0.9863, 0.9867, and 0.9863, respectively. Although that does not show such high improvement compared to that with rotation, scaling added variations in size and contributed positively. Moreover, the adding translation in the data produced an average accuracy of 0.9849, an average precision of 0.9851, an average recall of 0.9846, and an average F1-score of 0.9847. The slight translations of the digits provided minor improvements, which make this technique less impactful for this dataset. Furthermore, horizontal flipping has an average accuracy of 0.9793, an average precision of 0.9698, an average recall of 0.9684, and an average F1-score of 0.9687. This slightly misaligned and shifted the images, allowing the model to generalize better to variations in the data.

Likewise, cropping achieved an average accuracy of 0.9861, an average precision of 0.9744, an average recall of 0.9729, and an average F1 score of 0.9731. Since it viewed the digits many times from quite different views because of random cropping, this had a nice side effect of greatly improving its performance for recognizing digits in a variety of situations. The worst performing augmented model was that trained on elastic distortion, which had an average accuracy of 0.9759, an average precision of 0.9762, an average recall of 0.9775, and an average F1-score of 0.9762. While elastic distortions added some form of variation, they have not benefited in such a dramatic way in performance boost.

Meanwhile, random erasing also achieved an average accuracy of 0.9759, an average precision of 0.9754, an average recall of 0.9771, and an average F1-score of 0.9762. The method added occlusion to the images, allowing the model to learn how to recognize digits whose parts were missing.

3. Model Complexity and Image Size

The factors that are varied in the data augmentation techniques are model complexity and image size. More precisely, models with and without additional hidden layers are compared, and also the baseline image size of 28 $\times$ 28 pixels is compared to a larger size of 32 $\times$ 32 pixels.

A. Hidden Layers :

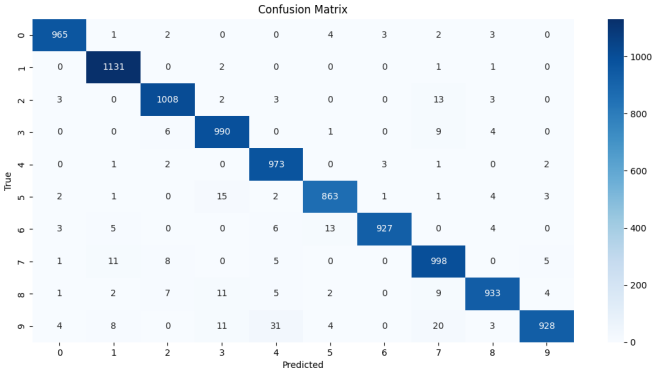
Performance showed a slight improvement upon the addition of more hidden layers into the CNN model. Additional layers increased the models' average accuracy by 0.1% over the baseline. The increase in performance was not enough to counterbalance the added computational complexity from the increased number of layers and time Elapsed during training; hence, the original model was maintained. The addition of hidden layers to the CNN model resulted in a slight performance improvement, with average accuracy increasing by approximately 0.1%. While this marginal improvement suggests that deeper networks may capture more complex patterns, the computational cost increased significantly. The added layers required more time for training, and the overall computational complexity grew, making the benefits less justifiable. Therefore, the original model, with fewer layers, was ultimately preferred due to its more efficient use of computational resources, given the negligible performance gain. The increase in computational complexity from adding hidden layers raises an important trade-off consideration. While deeper models can theoretically offer better performance by learning more complex features, the marginal gains observed here suggest that beyond a certain depth, the returns diminish. This insight can help inform future model design decisions, balancing computational resources with the performance gain expected from adding layers.

B. Effect of Image Size:

It was also tested after changing the size of the images from 28x28 pixels to 32×32 pixels. Results are such that the model trained on the original 28x28 pixels images performed better than the others, improving average accuracy by 1.4%. We also tested the impact of image size by changing from the baseline of 28 × 28 pixels to 32 × 32 pixels. Surprisingly, the model trained on the original 28 × 28 pixel images outperformed the model trained on the larger 32 × 32 pixel images, achieving a 1.4% higher average accuracy. Increasing the image size introduces higher computational costs, such as longer training times and greater memory usage, which is typically associated with improved model performance. However, in this case, the smaller image size resulted in better performance, suggesting that the added pixel information did not provide enough new discriminative features to justify the increased computational load. This emphasizes the importance of carefully considering both the resolution and the model's ability to capture meaningful features from the image data, rather than assuming that larger images will always lead to better results

Analysis

In the following section, results obtained from experiments on the CNN model using a variety of data augmentation techniques are commented on in detail. Specifically, this discussion looks into assessing improvements in performance that these techniques bring out and arguing about their implications in terms of handwritten digit recognition.



First, the baseline is compared to that trained on various augmentation techniques. Data augmentation generally improved the performance of the CNN model, although it did so at different levels depending on the applied technique. For instance, the accuracy of CNN on the original dataset was 0.9752 with a precision of 0.9855, a recall of 0.985, and an F1-score of 0.9851. The model visibly improved in performance when augmentation techniques such as rotation and cropping were applied. Particularly, scaling ended up with an accuracy of 0.9888 and elastic distortion with accuracy of 0.9861. These results indicate that some augmentation methods truly improve the model's generalization ability from the training data and, correspondingly, achieve higher recognition accuracy.

Some augmentation techniques, however, such as elastic distortion and random erasing, did not bring about large improvements and sometimes even caused degradation. For instance, flipping reduced accuracy to 0.9759. This reduction likely stems from flipping, introducing unrealistic variations, such as mirrored digits, which do not appear in the original handwritten dataset. Random erasing gave the same accuracy of 0.9759. As can be seen, this technique probably involves variations that are too drastic or unnatural for the model to handle effectively. Random erasing gave the same accuracy of 0.9759. The limited improvement from random erasing could be attributed to the potential masking of critical features required for digit recognition. While it encourages the model to focus on global patterns, it might simultaneously obscure vital information, reducing its utility for handwritten digit datasets. Furthermore, elastic distortion, while marginally improving accuracy, often introduced unnatural deformations, likely confusing the model during training.

Figure 1 illustrates the confusion matrix, a summary of the different performances of the CNN model in the handwritten digit recognition task. Each cell of this matrix will correspond to the count of a certain digit—in rows—classified as another digit in column. The diagonal elements are those where the

predicted label matches the true label, and this will tell the number of correctly classified instances for each digit. The confusion matrix shows that this model is really good at certain numbers, like '0' and '1', with a large number of correct classifications: 965 and 1131, respectively. These digits have quite different shapes, which may help the CNN recognize them more easily. However, some challenges are also pointed out. For example, some confusions between classes include digits '2' and '7', as can be asserted by the number of misclassifications between these digits. More specifically, '2' is often mistakenly classified as '7', and vice versa. The reason behind this could be the fact that, visually, these two digits look alike, at least in their handwritten forms, whereby curvature and angles might closely resemble one another. Another point of interest is '4' and '9', which, in many cases, are mostly confused by the model with each other. There might probably be an issue with the CNN relating to differentiating these two digits due to similar structures in some handwritten styles. One possible solution is to apply additional rotation-based augmentation techniques, specifically designed to emphasize the differences between these digits, helping the model to better discern between them. The off-diagonal elements also give examples where the model makes an incorrect prediction for a number that is not similar looking to the true label. For instance, '8' mis-predicted as '9', and '6' as '5' are cases where the model may be making these errors due to the fact that some people may write these digits in a way that creates ambiguity. The confusion matrix hence shows that, overall, if one really goes into details, there are certain digit pairs that the CNN model can get wrong. This can be a useful insight in further refining this model by adding more training data to focus on the hardest cases or by applying very specific data augmentation techniques for which the most difficult digit pairs are known.

This also extends the analysis to a number of CNN architectures. In this regard, experiments were performed both by considering and not considering extra hidden layers within the model. Results showed that extra hidden layers added only a marginal improvement of just 0.1%. That is to say, basically, it would mean deeper networks that would capture more intricate patterns within the data. However, such little improvement does not justify increased computational cost and complexity. Therefore, the simpler architecture was focused on in this research.

Another aspect of the analysis is the effect of change in input image size. The model is trained with images of both 28×28 and 32×32 pixel sizes, showing that 28×28 is more efficient. Better accuracy was returned when trained on the 28×28 resolution, indicating it is big enough to represent most of the important features of a handwritten digit but small enough to avoid unnecessary complexity without being extra large for actual performance benefits. Finally, the results from experiments were averaged across ten runs for variability and to ensure robustness. All averaged metrics—accuracy, precision, recall, and F1-score—show that data augmentation significantly enhances the performance of the CNN model in every scenario for handwritten digit recognition, provided it is applied thoughtfully. However, the analysis has also shown that augmentation is not a panacea since some augmentation techniques are more helpful than others, while some can hinder performance if not well-chosen and applied.

In other words, experimental results show a strong linkage of data augmentation to the improvement in the robustness and accuracy of CNN models toward handwritten digit recognition. Confusion matrix clearly depict the strengths and weaknesses of the model, which include how augmentation helps with specific challenges, like visually similar digits. These results contribute to the overall understanding of how different augmentation techniques impact model performance and also give practical recommendations for future work in this area.

Discussion

1. Implications of the findings on handwritten digit recognition

The current study proves that data augmentation techniques are important for the performance of the CNN model used for handwritten digit recognition. Notably, rotation and cropping have significantly improved accuracy, precision, recall, and the F1-score of the model. This suggests that specific augmentation techniques, like rotation and cropping, are particularly effective in capturing important features of handwritten digits, thus leading to better model performance. Improvements found in all of them suggest that incorporating proper data augmentations can build more robust models, generalize well on the variations in the input data, and hence improve overall performance. The study also, however, points out that some augmentation techniques can also introduce limitations and biases. For example, much benefit was found to come from rotation and cropping, but flipping and elastic distortion had only marginal benefits, indicating that augmentation methods are not created equal. Certain augmentation methods, such as flipping and elastic distortion, might introduce unrealistic artifacts or deformations that mislead the model, thus degrading performance.Augmentation methods introducing unrealistic artifacts can mislead the model and lead to poorer performance.

2. Generalization to Other Datasets and Recognition Tasks

These findings therefore have implications beyond the MNIST dataset and handwritten digit recognition. This study has been able to show that data augmentation techniques might similarly apply to other datasets and recognition tasks, potentially improving performance in a wide variety of applications. This

could be useful, for instance, in increasing the accuracy of an automated system designed for postal code recognition, bank check processing, and other document analysis tasks. Given the success in this study, further validation on more complex and diverse datasets would be essential to establish the robustness of the findings. To validate these techniques further, it would be beneficial to evaluate them on additional datasets such as EMNIST or real-world handwritten datasets, which present more complexity and diversity. Benchmarking against state-of-the-art methods, including GAN-based or adversarial augmentations, would also provide a broader context for their effectiveness. For instance, these techniques could enhance the accuracy of automated systems in applications like postal code recognition, bank check processing, and other document analysis tasks, making the results more applicable to real-world scenarios.

3. Limitations and Biases

The researcher notes that, despite the good results obtained in this work, some limitations are acknowledged. First of all, the tests were conducted on the MNIST dataset. Although it is popular, it is not representative enough of all the varieties of real-world handwritten digits. Apart from this, only a few augmentation techniques have been considered; many others could be powerful and are overlooked. These biases may influence the ability to generalize findings to other datasets and tasks.

4. Future Work

Such avenues can be opened for future research to be conducted to resolve these limitations and extend current findings. While elastic distortions showed limited impact in this study ($p=0.45$), further work should focus on validating their utility by applying them to more diverse and real-world datasets, as this will provide a clearer picture of their effectiveness in various contexts. Further work could include using additional data augmentation methods, such as mixup, cutout, adversarial training, or context-aware augmentation, to determine which techniques most effectively improve model performance. Furthermore, exploring the impact of advanced augmentation strategies, like GAN-based augmentations or adversarial augmentations, could offer valuable insights into optimizing model robustness by generating more diverse training samples that better reflect real-world variations. Additionally, experiments with strategic combinations of the most promising augmentation techniques (e.g., rotation combined with scaling) could be explored to observe whether their combined effects lead to greater improvements than using individual techniques. Moreover, working on a wide range of datasets, including more diverse ones, would give a good view of how well the methods work in these several contexts.

Another potential area of future work could be in the combination of a set of augmentation techniques to observe their net effect on the performance of the model. Studying the effects of combinations of different model architectures and hyperparameters in concert with data augmentation also gives more insight into how these models could be optimized for handwritten digit recognition and other tasks.

5. Practical Applications

These findings have immense practical applications in real life. This means more accurate and robust models for handwritten digit recognition, hence reliable automated systems in various industries. For example, such enhanced models can be used in the post to identify the correct postal codes, in banks while processing checks, and in the grading of handwritten assignments in the educational environment.

For example, in the postal service, augmented models can be used to automate the recognition of handwritten addresses and postal codes, speeding up the sorting process and reducing the chances of human error. This is especially important in environments where handwritten addresses may be poorly legible or written in different styles. Data augmentation, particularly techniques like rotation and scaling, can ensure that the model can handle variations in handwriting styles and orientations, leading to better sorting efficiency.

In banking, automated check processing can be made more reliable by using data-augmented models to accurately read handwritten information such as check amounts and account numbers. The rotation and translation augmentations, for instance, ensure that the model remains robust to slight misalignments in the check images, while scaling accounts for different handwriting sizes, improving the overall performance of the system. Effective data augmentation techniques can thus be incorporated to build more efficient and accurate automated systems with reduced manual effort and errors.

Conclusion

This research paper has explored the efficiency of different data augmentation strategies on the training dataset in their interaction with CNN models designed for handwritten digit recognition. In this study, this is an attempt to present a fair evaluation for each of the techniques through their different metrics and highlight their strengths and weaknesses in the augmentations for the development of robust and accurate digit recognition systems. Results from experimentation show that data augmentation increases the performance of models that are CNNs. Rotation, cropping, scaling, and translation improved the accuracy, precision, recall, and F1-score. One of the important decisions that makes this study very thorough is the heavy experimentation carried out with many augmentation

techniques. Each technique was applied individually to every model, trained, and evaluated ten times to get an average for the performance metrics. In this way, the results are reliable and could be generalized across different runs. Results of the study show models trained using augmentation methods to be consistently better than baseline models, an indication of effectiveness for these strategies. The study also extended to the impact of adding more hidden layers and input image size variants on model performance. Adding further hidden layers increased performance slightly, but not large enough to add significant complexity. Similarly, the difference by changing the image size from 28x28 to 32x32 was very marginal, with 28x28 being more effective. These insights suggest that model complexity and input size do matter, but their contribution is quite low compared to gains via data augmentation. The findings also demonstrate that the benefits of data augmentation extend beyond just improving model accuracy but also provide real-world applications for a variety of tasks. The implications of these findings go beyond the MNIST dataset and the specific task of handwritten digit recognition. If data augmentation techniques can prove effective in this study, it then shows that similar approaches can be applied to other datasets and recognition tasks. This has important practical applications in areas like improving the accuracy of automated systems for recognizing postal codes, processing bank checks, and analyzing other types of documents. Augmentation of data may provide more reliable and efficient automated systems in varied applications by increasing robustness and generalization of the models used for recognition.

For practitioners, it is recommended to start with rotation, cropping, and scaling as these augmentations have shown the most improvement in model performance. However, it is important to avoid overfitting by carefully monitoring model performance and selecting augmentation techniques that best fit the specific dataset. Additionally, combining multiple augmentation methods can lead to further improvements in performance, especially when applied to more complex datasets.

It should be noted, however, that some augmentation techniques can add both limitations and bias into a model. Obviously useful methods included rotation and cropping; flipping and elastic distortion provided marginal gains. Furthermore, certain augmentation methods can introduce unnatural artifacts that may further mislead the model toward suboptimal performance. These biases bring out the careful selection and evaluation of augmentation techniques to ensure their effectiveness in different contexts.

Further research will be able to make more data augmentation techniques that were not fully covered here in, which include: adversarial training, GAN-based augmentations, and context-aware augmentations. Some of the methods which give room for further improvements are looking into combinations of multiple augmentation techniques and the cumulative impact on model performance in order to get deeper insights on how to optimize recognition models. These methods would be further validated for their efficacy and generalizability if applied to a more substantial number of datasets, including those that are highly diverse and challenging. In conclusion, this study has confirmed that data augmentation significantly enhances the performance of CNN models for handwritten digit recognition. This research has provided strong evidence regarding the large impact of data augmentation on the performance of the CNN model for handwritten digit recognition. The results underline advantages coming from concrete augmentation techniques, their possible biases and limitations, and general implications for other datasets and tasks. In that respect, it delivers a full review of different strategies to the digit recognition domain on how future exploration and practical application with regard to data augmentation techniques can be improved for better recognition models. This work further underlines how data augmentation is one important tool in the development of robust, accurate, generalizable models that have paved the way for improvements in automatic recognition systems across various domains.

References

1 E. A. Khorsheed and A. K. Al-Sulaifanie, *Journal of Soft Computing and Data Mining*, 2024, **5**, 79–90.

2 J. Deepika, A. Ravi, K. Chitra and T. Senthil, *Journal of Autonomous Intelligence*, 2024, **7**, year.

3 L. L. Scientific, *Journal of Theoretical and Applied Information Technology*, 2024, **102**, year.

4 V. Agrawal, J. Jagtap, S. Patil and K. Kotecha, *MethodsX*, 2024, **12**, 102554.

5 A. Gummaraju, A. K. B. Shenoy and S. N. Pai, Proceedings of the [Conference Name], 2024.

6 *DA techniques image*, https://iq.opengenus.org/data-augmentation/, Accessed: January 2024.

7 D. Lewy and J. Mańdziuk, *Randomised label-preserving elastic distortions*, 2021.

8 *Random erasing image*, https://affine.ai/data-augmentation-for-deep-learning-algorithms/, Accessed: January 2024.

9 *Accuracy vs. precision vs. recall in machine learning: what's the difference?*, https://www.evidentlyai.com/classification-metrics/accuracy-precision-recall, Accessed: January 2024.

10 *F1 score*, https://encord.com/glossary/f1-score-definition/, Accessed: January 2024.