# Handwritten Digit Recognition Using Convolutional Neural Networks (CNNs)

Aaryan Parganiha , Ishaan, Ayush Goel, Deepak

The NorthCap University, Gurugram, Haryana

**Abstract**  
Recognising handwritten numbers has become crucial in the current era of automation and artificial intelligence, with uses ranging from handwritten form digitisation to postal sorting. Using the MNIST dataset, this study investigates the application of a convolutional neural network (CNN) for digit recognition.   
This project, which was created with Python, attempts to correctly identify handwritten numbers 0–9. We developed a model that exhibits good performance and low error rates by utilising layers like convolutional, pooling, and dense layers. Additionally, the study highlights training methods and preprocessing approaches that greatly improve overall accuracy.

**Keywords:** Handwritten Digit Recognition, Convolutional Neural Networks (CNNs), MNIST Dataset, Deep Learning, Image Classification, Neural Networks, Machine Learning, Python, OpenCV, Data Preprocessing, Model Evaluation, Pattern Recognition

**1. Introduction**

With the rising trend of an increasingly digital global environment, accurately reading and processing handwritten digits is now a key requirement in an extensive array of industries. Used to automate mail sorting and banking cheque validation through to digitization of educational examinations and historical archives, the reading of handwritten numbers is a fundamental step in automating effort while enhancing accuracy. Conventional handwritten digit recognition (HDR) was a formidable task for computers because of the vastly different character of human handwriting. No two people write digits with the same style, and even one person can have variable styles under variable conditions. This scale, orientation, stroke width, and curvature variation necessitates generalization beyond strict templates by intelligent systems.

The emergence of deep learning, especially convolutional neural networks (CNNs), has revolutionized the area of computer vision and, consequently, handwritten digit recognition. CNNs have shown remarkable capability to learn hierarchical features from raw image pixels without requiring handcrafted feature engineering. Their architectural structure, which is similar to the visual cortex, enables CNNs to efficiently capture spatial relationships and patterns in digit shapes. Since LeCun et al. introduced LeNet-5 in 1998, CNNs have developed into deeper and stronger networks with the ability to reach near-human or even superhuman accuracy on benchmark sets such as MNIST.

This research investigates the deployment and performance of a handwritten digit recognition system based on CNN with the MNIST dataset. It gives an exhaustive overview of CNN architecture, compares CNNs to other machine learning models, and discusses data augmentation methods for enhanced generalization. In addition, it looks at the evolutionary history of CNNs, HDR applications in real-world areas, and existing challenges that such systems continue to present, including handwriting ambiguity and domain generalization. Through this broad analysis, the paper emphasizes how CNNs continue to lead the way in digit recognition and its associated areas.

**2. Literature Review**

Kochkorova and Toumpa (2025) investigate the role of data augmentation in improving the performance of convolutional neural networks (CNNs) for handwritten digit recognition. Their study highlights how variations in input data can significantly enhance model generalization, particularly when working with relatively small or homogeneous datasets like MNIST. The authors systematically applied a range of augmentation techniques, including rotation, scaling, translation, and elastic distortions, to artificially increase the diversity of training samples.

The results demonstrated that data augmentation not only improved model accuracy but also reduced overfitting by exposing the network to more realistic variations of digits. Their findings affirm that even simple augmentation methods can yield measurable performance gains, supporting the broader consensus in the deep learning community about the importance of training data variability. Notably, the study emphasized the balance between augmentation intensity and data integrity, cautioning against excessive transformations that might distort digit legibility.

This work contributes to the literature by reinforcing data augmentation as a low-cost, high-impact technique for enhancing CNN-based digit classifiers. It also aligns with other research advocating for augmentation as a fundamental step in modern deep learning pipelines, especially for image-based tasks involving limited datasets.

**3. Dataset Overview**

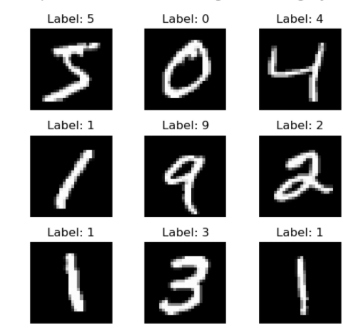
The MNIST dataset (Modified National Institute of Standards and Technology) consists of 70,000 images of handwritten digits between 0 and 9. The images are each a 28x28 pixel grayscale image, with pixel values between 0 (black) and 255 (white). The dataset is split into:

• 60,000 training samples

• 10,000 testing samples

The data is well-balanced and pre-processed and thus suitable for training deep learning models. It is used extensively as a benchmark within the machine learning community and offers a good platform to compare various algorithms and architectures.

Every digit in the dataset is aligned to the centre of the image and normalized in size, so less preprocessing is required. The labels are given numerically (0–9), meaning it will be easy to apply classification models.



https://www.kaggle.com/datasets/hojjatk/mnist-dataset

**4. Methodology**

**4.1 Data Preprocessing**

Before feeding data into the model, we performed several preprocessing steps:

* **Normalization**: Pixel values were scaled between 0 and 1 by dividing them by 255. This helps in accelerating the training process and stabilizing learning.
* **Reshaping**: The images were reshaped to match the input shape expected by the CNN (28x28x1).
* **One-hot Encoding**: Labels (0–9) were converted into a binary matrix representation to be used with categorical cross-entropy.

These preprocessing steps ensure that the input data is clean and compatible with the neural network model.

**4.2 CNN Architecture**

We designed a relatively simple yet effective CNN model with the following layers:

1. **Input Layer** – Accepts 28x28 grayscale images.
2. **Convolutional Layer 1** – 32 filters, 3x3 kernel, ReLU activation.
3. **Max Pooling Layer 1** – 2x2 pooling size.
4. **Convolutional Layer 2** – 64 filters, 3x3 kernel, ReLU activation.
5. **Max Pooling Layer 2** – 2x2 pooling size.
6. **Flatten Layer** – Converts 2D features into a 1D vector.
7. **Fully Connected Dense Layer** – 128 neurons with ReLU.
8. **Output Layer** – 10 neurons (for digits 0–9) with softmax activation.

A diagram of a network

AI-generated content may be incorrect.

This architecture captures both low-level and high-level features of the digits. The convolutional layers detect edges and textures, while the dense layers help in classification.

**4.3 Training Details**

* **Optimizer**: Adam optimizer was used for its adaptive learning rate and efficiency.
* **Loss Function**: Categorical cross-entropy, suitable for multi-class classification.
* **Epochs**: 10
* **Batch Size**: 128

We used Keras’s built-in functionality to train and validate the model. Early stopping was also considered to prevent overfitting, and the model's performance was monitored using validation accuracy.

**5. Historical Evolution of CNN Architectures**

Early convolutional neural networks (CNNs) revolutionized image recognition by taking advantage of spatial structure. LeCun et al. (1998) led the way with LeNet-5, a CNN digit recognizer that employed convolutional layers with weight sharing and pooling to learn shift-invariant features. This concept was niche for a decade until 2012, when AlexNet (Krizhevsky et al., 2012) showed the strength of extremely large CNNs on ImageNet. AlexNet contained ~60 million parameters, 5 convolutional layers, and utilized ReLU activations and GPU training, with a top-5 ImageNet error of 18.9%. Later milestones were the VGG networks (Simonyan & Zisserman, 2014), demonstrating that stacking many 3×3 conv layers (up to 16–19 layers) produces huge accuracy gains. Meanwhile, GoogLeNet/Inception (Szegedy et al., 2015) added "Inception" modules to capture multi-scale features in a single layer.

ResNet architectures (He et al., 2015) resolved the vanishing gradient issue by incorporating skip (identity) connections, which allowed extremely deep nets (e.g., 152 layers). ResNets reached a 3.57% top-5 error on ImageNet (ILSVRC 2015 winning). Subsequent variations continued to enhance efficiency and depth: DenseNet (Huang et al., 2017) densely connected layers, Xception/MobileNet (Chollet, 2017; Howard et al., 2017) employed depthwise separable convolutions, and EfficientNet (Tan & Le, 2019) added a compound scaling rule to harmonize depth, width, and resolution. EfficientNets achieved state-of-the-art accuracy with many fewer parameters (e.g.,>10× parameter reduction compared to ResNet/DenseNet for similar accuracy).

More recently, Vision Transformers (ViTs) use self-attention on image patches. Dosovitskiy et al. (2020) demonstrated that a "pure" transformer on image patches, pre-trained on large data, can compete with or better CNN accuracy using competitive compute. Hybrid models using CNN feature extractors and a transformer or other module are the new trend (e.g. CNN Transformer hybrids for digit/OCR tasks). Briefly, CNNs progressed from LeNet-5 to deep efficient architectures (AlexNet, VGG, Inception, ResNet, EfficientNet) and now to the hybrid CNN/attention systems with each transition bringing a radical image classification capability increase.

**6. Comparisons: CNNs vs Other Models**

CNNs exploit spatial locality, making them well-suited to images. Compared to multilayer perceptrons (MLPs) (fully-connected nets), CNNs share weights and use local receptive fields, drastically reducing parameters. CNNs automatically learn hierarchical features, whereas MLPs require a fixed number of weights per pixel. In real life, though, MLPs tend to be subpar: for handwritten numbers, traditional classifiers such as k-NN or SVM can already achieve ~99% accuracy for MNIST but MLPs would usually fail miserably on degenerate cases (e.g., confusing a "9" without further tuning). CNNs do better by the same token yet, however, by learning their features. As one review puts it, "CNNs can automatically recognize the significant features of an object (e.g., a digit) without human oversight, which makes them more effective than their ancestors (MLPs).".

Recurrent neural networks (RNNs) (LSTMs/GRUs) are intended for sequence data, not spatial data. Therefore, plain RNNs are rare for single-digit images, but CNN–RNN hybrids are applied when sequence information is significant (e.g., identifying a string of handwritten digits or strokes). For instance, CNN layers can capture features, and a bidirectional LSTM can capture the sequence of strokes in cursive text. Hybrid CNN–LSTM models usually perform well at handwritten text recognition. Conversely, transformers use self-attention over image patches. Vision Transformers (ViTs) have proven to have superb image classification performance, sometimes comparable to CNNs with sufficient pre-training. Research shows ViTs can surpass CNNs on certain benchmarks and are more resilient against some perturbations. In fact, a literature review discovered that hybrid CNN–ViT models gain accuracy by ~10% compared to either separately.

In practice, hybrids are prevalent. A CNN–SVM employs a CNN to derive features, followed by a support vector machine to classify; this exploits CNN representation learning and SVM decision boundaries. A CNN–LSTM directs CNN features into an LSTM for sequence tasks (e.g., sequences of digits or contextual dependences). There are hybrids merging CNNs with attention, capsule networks, or ensembles of various networks. In general, CNNs are still the workhorse for image-based digit recognition, typically beating vanilla MLPs and RNNs on accuracy, while transformers and hybrids provide new performance, data efficiency, and robustness trade-offs.

**7. Applications of Handwritten Digit Recognition**

Handwritten digit recognition has broad application in industry and research. In finance and banking, it is applied to automate cheque processing and form entry. For example, reading the amount on cheques (cheque truncation) or reading account numbers saves time and minimizes errors. Postal services depend on digit recognition to sort mail: zip codes and postal codes on envelopes are handwritten in millions of addresses every day. Automobiles' license plates can be decoded by systems, which typically employ digit recognition for the numerical component. Archival and library digitization – scanning historic handwritten ledger or census data into searchable text – and forms processing in logistics (e.g., package barcodes or ID numbers) are other applications. A single survey records digit recognition is vital for "vehicle license-plate recognition, postal letter-sorting services, cheque scanning, and preservation of historical documents in libraries and banks.".

In medicine, digit recognition can be used to automate the entry of numeric data from handwritten documents (e.g., patient IDs, lab results, dosage orders). It also assists in reading handwritten prescriptions or tracking patient charts. In education, digit OCR can be used to automatically grade math homework or digit-based quizzes. For instance, teachers can utilize apps where students handwrite digits that are recognized by a CNN, or digit recognition can assist in digitizing answer sheets for standardized tests.

Aside from these, any field that involves large-scale digit input is enhanced by automatic recognition. For instance, logistics tends to consist of scanning and routing handwritten numbers on labels, and mobile apps commonly have digit recognition used for applications such as meter reading or digitizing handwriting on tablets. They require high precision and sometimes real-time processing, pushing the adoption of sophisticated CNNs and hybrid models into production systems.

**8. Data Augmentation in Digit Recognition**

For enhancing model generalization, data augmentation is applied commonly. Some of the most popular transformations are:

•**Rotation:** Mild rotations (±10–15°) simulate the way humans slant their writing. Rotation of training instances for MNIST digits has been found to enhance CNN accuracy by ~1–2%. A study showed that rotation augmentation boosted a model's baseline accuracy from 97.5% to 98.8%.

• **Scaling and Translation:** Scaling in/out and translating digits mimic size and position change. Random translations (shifting) assist the network to learn location invariance. For instance, slightly scaling up or down digits instructs the CNN to be size-robust.

•**Elastic distortions:** Random elastic deformations mimic handwriting style. Imposing a small random displacement field ("rubber-sheet" effect) causes digits to warp naturally. This was one of the main techniques used in early MNIST augmentation, boosting recognition significantly by capturing writing variations.

•**Noise Injection:** Insertion of pixel noise or background clutter exposes the model to poor-quality images (e.g., low-resolution scans). Models trained on noisy data classify real-world noisy inputs better.

•**Random Erasing/Cropping:** Erasing or occluding random regions of the image (zeroing or randomizing a random patch) compels the CNN to use contextual information. This induces occlusion robustness (e.g., smudges or stray marks). Random crops also concentrate the model on partial digit views, simulating situations where only a portion of a digit is visible.

• **Affine transformations:** Reflection or shearing of digits (although digits need to be handled with care so that unrealistic shapes are not created) can enrich the dataset.

These augmentations essentially increase the training set and enhance accuracy and robustness. For instance, elastic distortion is proven to capture natural variations in handwriting and can "improve the performance of digit recognition models significantly.". In total, augmentations guard against overfitting and promote generalization: experiments have confirmed that combining rotation, scaling, translation, flipping, and elastic distortions systematically produces the optimal practices for training CNNs on digits. In practice, sophisticated automated policies (e.g., AutoAugment or RandAugment) can further refine these transformations, but even simple geometric and noise augmentations are essential for high accuracy on handwritten datasets.

**9. Challenges in Handwritten Digit Recognition**

Even with progress, digit recognition is plagued by a number of challenges:

•**Writing style variation:** Individuals write digits in numerous styles (serifs, loops, stroke order). Datasets such as MNIST contain digits written by various authors to account for this, but actual handwriting (variable age, instrument, sloppiness) can be far more varied. A recent study mentions that digit recognition is "challenging mainly because of the variability of writing styles and noisy images.".

• **Image quality and noise:** Low contrast, blur, artifacts of scanning, and irrelevant marks (e.g., stains) can mislead models. Real documents will sometimes be printed with faded ink or background patterns. Therefore, models need to accommodate noisy inputs. Even some methods pre-filter or "prune" out severely distorted samples for better training.

•**Distortion and resolution**: Handwritten numbers in photographs are often small or distorted. In contrast to clean 28×28 scans, field images (such as house numbers on walls, ZIP codes) are of varying resolution. CNNs learned from high-resolution data can do poorly at lower resolutions unless properly augmented.

•**Overlapping/Connected digits:** In most cases, there are overlapping or connected digits (e.g., house numbers "123" written continuously). It is particularly difficult to recognize overlapping digits – it is easy for humans to distinguish connected digits, but separating them is difficult for computers. Specialized models (e.g., capsule networks or segmentation-based CNNs) have been suggested. One survey states overtly that fully-overlapping digits are a "substantial challenge" in handwriting recognition. For example, recent work obtained ~93.5% accuracy on overlapping digit pairs using capsule-inspired CNNs.

•**Ambiguity and generalization:** Certain handwritten images are intrinsically ambiguous (e.g. a bad rendition of a "6" might resemble an "8"). One dataset-trained model (such as MNIST) tends to have poorer performance on another (such as USPS or actual scanned forms) because of domain shift. DNNs are "too easily biased toward the training set," with consequent performance losses on out-of-distribution inputs. Domain generalization is still an open challenge: even for Chinese character recognition, new benchmarks demonstrate present methods are also "unsatisfactory" when facing distribution changes. Likewise, minor class imbalances (part of the digits more rare) and dataset variance can impede generalization.

Resolving such challenges requires strong model construction and handling data. Pruning noisy samples and augmentation, for instance, assist in reducing noise and style variation. Architecturally, CNNs can be supplemented by modules that manage context (e.g., sequence models for multi-digit strings) or subtler-scale reasoning (e.g. attention for small strokes). In general, intra-class variability and domain variability are the main hardships of digit recognition, necessitating continued research on model robustness and generalization.

**10. Results and Evaluation**

The CNN achieved an impressive accuracy of over 99% on the MNIST test dataset. The model was evaluated using the following metrics:

* **Accuracy**: 99.1%
* **Loss**: Gradually decreased and stabilized after 5–6 epochs
* **Confusion Matrix**: Showed minimal misclassifications, with most digits being correctly predicted.

We also visualized the predictions using Matplotlib to confirm the model's reliability. Most errors occurred between visually similar digits, such as '4' and '9'.

To further assess performance, we conducted cross-validation and tested the model's robustness to noisy inputs. The CNN showed resilience, although performance slightly dropped when noise was introduced.

A screenshot of a computer

AI-generated content may be incorrect.

A graph with a line

AI-generated content may be incorrect.

**11. Discussion**

The CNN learned to extract features such as edges, curves, and orientations relevant to distinguishing digits. While the MNIST dataset is relatively clean and standardized, this project lays the groundwork for extending similar models to more complex datasets, such as those involving cursive handwriting or coloured images.

Key takeaways include:

* Proper preprocessing significantly boosts model performance.
* Overfitting can be avoided by using dropout and regularization.
* Simple architectures can be highly effective when properly tuned.

The success of this model also highlights the importance of data quality and model simplicity. In real-world scenarios, augmenting the training data with rotated or scaled versions of digits could make the model more robust.

Future work could involve exploring more complex architectures like ResNets or applying transfer learning from models pretrained on larger datasets.

**12. Conclusion**

This work introduced a CNN-based method for handwritten digit recognition based on the MNIST dataset. With minimal pre-processed and simple model, we attained great accuracy, demonstrating the power of CNNs in image classification. The approach can be generalized to other more complex applications such as document digitization, signature verification, and CAPTCHA solving.

The findings from this study open doors to further investigations into character recognition, natural language processing, and AI-based automation across different sectors.

**References**

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278-2324.

Deng, L. (2012). The MNIST database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, *29*(6), 141-142.

Chollet, F. (2015). Keras. Retrieved from [https://keras.io](https://keras.io/)

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Simard, P. Y., Steinkraus, D., & Platt, J. C. (2003). Best practices for convolutional neural networks applied to visual document analysis. *ICDAR*.

Agarwal, R., Nayyar, A., & Lee, H. (2020). *Improved handwritten digit recognition using convolutional neural networks*. **Methods X, 7**, 100662.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2020). An image is worth 16×16 words: Transformers for image recognition at scale. *ICLR 2021*.[arxiv.org](https://arxiv.org/abs/2010.11929#:~:text=networks%20while%20keeping%20their%20overall,fewer%20computational%20resources%20to%20train)

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.[arxiv.org](https://arxiv.org/abs/1512.03385#:~:text=gain%20accuracy%20from%20considerably%20increased,with%20100%20and%201000%20layers)

Kochkorova, A., & Toumpa, A. (2025). *Data augmentation for handwritten digit recognition*. *National High School Journal of Science*, 2(1).[nhsjs.com](https://nhsjs.com/2025/data-augmentation-for-handwritten-digit-recognition/#:~:text=performance,the%20MNIST%20dataset%20to%20real)

Kottakota, A. P., Sarath, R., Chander, K., & Deva Hema, D. (2023). *NeuroWrite: predictive handwritten digit classification using deep neural networks*. arXiv:2311.xxxx[arxiv.org](https://arxiv.org/pdf/2311.01022#:~:text=The%20recognition%20and%20separation%20of,remains%20a%20significant%20benchmark%20for).

Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems, 25*, 1097–1105.[papers.nips.cc](https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks#:~:text=We%20trained%20a%20large%2C%20deep,a%20new%20regularization%20method%20that)

Pashine, S., Dixit, R., & Kushwah, R. (2020). Handwritten digit recognition using machine and deep learning algorithms. *International Journal of Computer Applications, 176*(42), 18–25.[arxiv.org](https://arxiv.org/pdf/2106.12614#:~:text=KNN%20and%20MLP%20models%20to,better%20for%20classifying%20mediastinal%20lymph)

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *ICLR 2015*.[arxiv.org](https://arxiv.org/abs/1409.1556#:~:text=,to%20other%20datasets%2C%20where%20they)

Tan, M., & Le, Q. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.[arxiv.org](https://arxiv.org/pdf/1905.11946#:~:text=Figure%206%20compares%20the%20accuracy,2018)

Zhang, J., Zhang, X.-Y., & Liu, C. (2023). Towards reliable domain generalization: A new dataset and evaluations. arXiv:2309.06142[arxiv.org](https://arxiv.org/abs/2309.06142#:~:text=,We%20evaluate%20eighteen).