

Bridging the Domain Gap in Air-Written Digit Recognition

Nitasha Thakur¹, Nandini Gupta², Dr. Sonia Sharma³, Ms. Shavi Juneja⁴

^{1,2}Students, Department of CSE, MIMIT Malout

³Professor & Head, Department of CSE, MIMIT Malout

⁴Lecturer, Department of CSE, MIMIT Malout

Abstract

Air-written digit recognition connects touchless interaction with image classification, but a model trained on conventional handwriting may not generalize to traces created from webcam-tracked finger movements. This study builds on an existing hand-tracking and virtual-painter application developed with MediaPipe, OpenCV and a CNN trained on MNIST. The prototype tracks the index fingertip, renders strokes on a digital canvas and displays a predicted digit with class confidence scores. The proposed research will investigate supervised fine-tuning using a custom air-written dataset. An initial collection target of 20 writers and 4,000 labelled images is defined, with writer-separated training, validation and test partitions. The evaluation design preserves the source-trained checkpoint and compares original preprocessing, standardized preprocessing and target-domain fine-tuning so that their effects can be distinguished. Accuracy, precision, recall, macro-F1 and confusion matrices will assess classification, while capture failures and recognition latency will describe practical system behavior. Existing project screenshots provide qualitative implementation evidence. Custom dataset collection and fine-tuning have not yet been completed, so performance gains are treated as hypotheses rather than experimental findings. The intended contribution is a reproducible methodology and application-specific dataset for determining whether adaptation improves recognition of unseen writers without confusing interface demonstrations, confidence scores and measured accuracy.

Keywords: Air-writing, digit recognition, domain adaptation, convolutional neural network, MediaPipe, virtual painter, fine-tuning.

I. INTRODUCTION

A. Background and Motivation

Air-writing allows a person to form a digit in free space by moving a finger in front of a camera. A computer follows the motion and converts it into a digital trace. This interaction is useful when a keyboard, mouse or drawing tablet is inconvenient, particularly for short classroom demonstrations and touchless numerical input. The present work focuses on a standard webcam and isolated digits from zero to nine. It builds on an existing hand-tracking and virtual-painter application developed as an undergraduate project. The physical act of writing changes when a supporting surface is removed. A pen leaves a persistent mark and provides friction, while a finger moving through the air has neither support nor a visible physical trail. The user instead depends on the screen for feedback. Small tracking errors, interrupted gestures and

changes in drawing speed can alter the digital shape. A recognizer trained on conventional handwritten images therefore receives an input that may differ from the examples used during its training.

Convolutional neural networks provide an established approach to learning image features for digit classification [1]. MNIST offers a convenient source of labelled examples, with 60,000 training images and 10,000 test images at a resolution of 28 by 28 pixels [2]. However, successful classification of MNIST images does not by itself establish reliable recognition of a webcam-generated drawing. This study investigates that practical distinction within the existing project rather than treating an offline benchmark as a complete measure of application quality.

B. Existing Project Foundation

The implemented application combines MediaPipe hand landmarks, OpenCV image processing and a virtual drawing canvas. The index fingertip controls the drawing position. Finger-state rules allow drawing and tool selection, while the canvas stores the accumulated strokes separately from the camera background. The recognition component accepts a cropped drawing and displays a predicted digit together with class confidence scores. The project includes both a desktop launcher and a web interface that launches local Python modules.

The students have trained the recognition model using MNIST. At this stage, a labelled multi-user air-writing dataset has not been collected and the model has not been fine-tuned on such a dataset. Consequently, the present manuscript develops a research methodology around the available prototype. The implemented interface provides the capture mechanism, and the proposed extension provides a controlled way to study recognition on air-written digits. The distinction between completed functionality and planned experimentation applies throughout the paper.

C. Problem Statement

The research problem is to determine whether adapting an MNIST-initialized CNN to labelled air-written images improves recognition for users who did not contribute training samples. The source and target inputs represent the same ten classes, but their strokes, spacing, curvature and acquisition processes differ. This difference in input distribution is referred to as a domain gap. The existence and size of the performance gap in this particular application must be measured; it cannot be inferred from a few successful demonstrations.

An additional difficulty is separating model errors from capture and preprocessing errors. A malformed image caused by accidental stroke joining may be misclassified even by a well-trained model. Conversely, a visually correct input may fail because its foreground polarity or scale differs from the model's expectations. A defensible comparison must therefore keep the capture process, preprocessing and test set consistent while changing the training strategy. Otherwise, an apparent improvement cannot be attributed to fine-tuning alone.

D. Research Questions and Objectives

The principal question is whether supervised fine-tuning increases macro-averaged F1-score on held-out air-written digits compared with an unchanged MNIST-trained CNN. Supporting questions concern which digits remain difficult, whether preprocessing alone changes performance, and whether adaptation affects response time. A further question is whether improvements are shared across writers or concentrated among a small number of familiar writing styles.

The objectives are to document the existing gesture interface, define a reproducible capture procedure, build a labelled dataset with writer identifiers, establish a frozen baseline and implement controlled fine-tuning. Evaluation will include accuracy, precision, recall, F1-score and a confusion matrix. The study

will also record capture failures and recognition latency so that classification quality is interpreted alongside the usability of the complete system.

E. Scope and Intended Contribution

The study is limited to one completed digit per recognition request. It does not claim continuous sentence recognition, sign-language translation or support for every user with a motor disability. The hand tracker remains a pretrained component. The planned learning experiment changes the digit classifier and uses images rendered from fingertip movement. A standard RGB webcam is the intended input device, so depth cameras, wearable sensors and external drawing tablets are outside the experimental scope.

The intended contribution is a documented air-writing dataset and a reproducible baseline-versus-adaptation experiment within a low-cost prototype. It is not a claim that air-writing or transfer learning is a new concept. A useful outcome may be a measured improvement, a limited improvement or evidence that preprocessing and capture quality dominate the task. Each outcome would clarify how a classroom-scale project should be evaluated before broader use.

II. LITERATURE REVIEW

A. Handwritten Digit Recognition

LeCun and colleagues demonstrated how gradient-based learning and convolutional networks can learn useful representations for document recognition [1]. This work supports the use of trainable image features rather than relying only on manually specified character rules. Its relevance here is the digit-classification foundation; it does not establish performance for the current webcam interface. The MNIST loading documentation also defines the data dimensions and train/test sizes used by the source-domain training workflow [2].

For this project, MNIST is a starting point because its labels match the intended output classes. It allows a ten-class CNN to learn edges, curves and digit structures before target-domain data are available. The benefit of that initialization is a hypothesis to test against suitable controls. A network trained from random initialization on the same target training set can help determine whether the source training provides value beyond merely increasing the total training effort.

B. Hand Tracking and Fingertip Capture

MediaPipe Hands describes a pipeline consisting of palm detection and hand-landmark estimation from RGB imagery [3]. The current Hand Landmarker documentation exposes image, video and live-stream modes, along with hand landmarks and confidence-related configuration options [4]. The project uses the video-mode API and consumes fingertip coordinates to build its canvas. These sources justify the tracking component, but their performance measurements cannot be transferred directly to the students' hardware. Mukherjee and colleagues studied fingertip detection and tracking for air-writing in videos [5]. Their work addresses writing-pose initialization and termination as well as fingertip localization. This is relevant because a recognizer needs a correctly delimited character, not merely a sequence of detected hands. The present project uses simpler finger-state rules and an explicit recognition button. Its research focus is adaptation of rendered digit images rather than reproduction of that paper's detection algorithm.

More recent work by Wu and colleagues introduces a video-based air-writing dataset and a recognizer that uses spatial and temporal information for Chinese characters [6]. It demonstrates that camera-based air-writing datasets and sequence models are active research directions. Its vocabulary and representation differ from the ten-digit image task proposed here. Accordingly, its reported performance is not used as a

numerical baseline for this project, and the custom dataset is positioned as application-specific rather than the first dataset of its kind.

C. Transfer Learning and Domain Adaptation

Pan and Yang discuss transfer learning when training and future data differ in distribution or task setting [7]. Wang and Deng review deep visual domain adaptation and its major categories [8]. These works provide terminology for the proposed change from conventional handwriting images to air-writing renderings. Here, digit labels remain the same and labelled target samples will be available, so the planned method is supervised target-domain fine-tuning.

Ganin and colleagues propose domain-adversarial training using labelled source data and unlabelled target data [9]. That approach offers a useful alternative research direction, but it introduces a domain classifier and an additional training objective. The current proposal does not implement adversarial alignment. A smaller supervised experiment is chosen because the team can obtain labels directly when participants draw prompted digits, allowing the effect of ordinary fine-tuning to be examined first.

The Keras transfer-learning guide explains freezing pretrained layers and later fine-tuning selected layers with a small learning rate [10]. It also highlights the need to recompile after changing layer trainability. These principles inform the proposed two-stage training procedure. Since both source and target tasks have ten identical digit labels, the existing output layer can be retained; a new label vocabulary or a large pretrained natural-image network is not required.

D. Evaluation and Confidence Interpretation

Group-aware splitting tools allow samples belonging to one group to remain together when creating partitions [11]. Writer identity is the relevant grouping variable in this study because multiple examples from one person share writing characteristics. A random image split could place very similar samples from the same writer into training and testing. The proposed writer-separated protocol therefore addresses a practical source of optimistic evaluation.

Classification-report tools provide per-class precision, recall, F1-score and support [12]. These quantities will accompany overall accuracy so that a weak digit class is not hidden by aggregate performance. Confidence scores require a separate interpretation: Guo and colleagues show that neural-network probabilities can be poorly calibrated and discuss temperature scaling [13]. The confidence graph in the prototype is therefore treated as a display of model scores, not a direct measurement of recognition accuracy.

E. Recent Studies from 2024 to 2026

Qedear et al. (2024) developed SamAbjd, an educational application for Arabic air-writing, and investigated CNN and VGG16 models with preprocessed handwritten-letter datasets [14]. Their work provides a recent example of combining fingertip interaction, image preparation and a recognition interface. For the present proposal, the relevant lesson is to evaluate the captured target-domain traces separately: performance on an image dataset should not be assumed to establish recognition quality for unseen air-writing users.

Feng et al. introduced RingRT online in 2025, with its journal volume published in 2026 [15]. The system uses an IMU-equipped ring and a CNN–Transformer architecture to recognize characters written within a palm-sized region. Its user-independent testing and multiple collection scenarios reinforce the importance of writer-separated evaluation. Because RingRT processes motion-sensor signals, its reported performance is not a directly comparable benchmark for this webcam-based digit classifier.

Nakamura and Miwa (2026) proposed Mobile-AeroText for Japanese Hiragana input [16]. Their method converts fingertip trajectories into binary images and uses a GELAN-based object detector to locate and recognize character regions without explicit boundary gestures. The study evaluates recognition, response time and user experience. It informs the proposed future extension to automatic writing boundaries and broader usability testing, while the initial experiment here remains restricted to isolated digits.

Imbert et al. (2026) investigate unsupervised domain adaptation for character classification and handwritten text recognition [17]. Their pipeline learns from labelled source data and unlabelled target data, using target-style source images to train the recognition model. This provides a recent alternative for situations with scarce target annotations. The present study instead proposes supervised fine-tuning on labelled air-written digits; comparing these adaptation strategies would require an additional controlled experiment, beyond the initial protocol.

F. Research Gap and Position of the Study

The gap addressed here is specific to the current application: the existing MNIST-trained model has not been evaluated on a controlled, writer-separated air-writing dataset, and no adaptation experiment has been completed. The literature establishes relevant methods, but it does not answer how this particular rendering pipeline, preprocessing choice and user population will interact. A local benchmark is needed before claims about practical improvement can be supported.

The study will compare the unchanged model with a fine-tuned version under identical input processing and evaluation conditions. Additional comparisons will isolate preprocessing changes and training from scratch where resources permit. This design connects the research question to an observable difference in held-out performance. It also keeps the proposed contribution modest: a reproducible experiment and documented dataset are more defensible than a general claim of universally superior recognition.

Table I: Research foundations and their role in this study.

Research area	Refs.	Role in the proposal
Digit CNNs	[1], [2]	MNIST source task and ten-class image input
Hand and fingertip tracking	[3]–[5]	Capture method and writing boundaries
Video air-writing	[6]	Related dataset and temporal-model direction
Domain adaptation	[7]–[10]	Supervised fine-tuning and alternative approaches
Evaluation	[11]–[13]	Writer grouping, metrics and score interpretation

III. CASE STUDIES

A. Existing Virtual Painter as a Capture Platform

The first case examines the prototype as a practical tool for producing digit images. A user raises the index finger, moves it through the desired shape and watches the resulting stroke on the screen. OpenCV connects successive drawing positions on a persistent canvas. Color selection and erasing are available, making the interface useful for demonstrating gesture control. For dataset collection, however, these flexible controls must be restricted so that images are produced under a documented policy.

The report includes screenshots showing hand landmarks and virtual drawing. Figure 1 reproduces an existing hand-tracking view, while Figure 2 shows a drawing example. These images provide qualitative

evidence of the interface and its visual feedback. They are not a test set and do not establish detection accuracy, frame rate or recognition reliability. Their purpose is to show the implemented foundation on which the proposed collection workflow will be built.

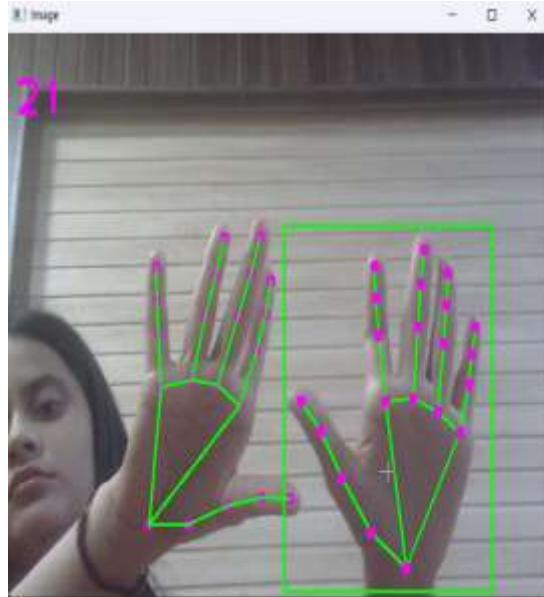


Figure 1: Hand-landmark display from the existing project report.



Figure 2: Virtual drawing in the existing prototype. Toolbar appearance reflects the report version.

B. MNIST Input and Air-Written Input

A digit drawn on paper and a digit rendered from tracked coordinates can represent the same class while differing in shape. A curved stroke may become angular when few camera frames are available. A thick brush may close a loop that would remain open in a thinner trace. A user may pause midway through a digit, and an incorrect resume operation may join separate strokes. These changes arise before the CNN sees the input.

The proposed investigation will inspect raw canvas snapshots and their normalized 28 by 28 versions together. If the raw image is recognizable but the normalized image loses important structure, preprocessing is the likely cause. If both are clear and the model still predicts incorrectly, the classifier

becomes the more plausible source of error. This simple distinction prevents every failure from being attributed to a domain gap without examining the input pipeline.

C. Writer Variation

Different writers form digits differently even when asked to follow the same task. A seven may include a crossbar, a four may be open or closed, and a one may contain an initial slant or a base. Air-writing adds variation in arm movement, preferred drawing size and speed. A dataset collected from only one person can therefore produce an apparently successful model that works mainly for that person's style.

The proposed collection plan uses multiple writers and keeps their samples together during partitioning. Training writers supply examples for optimization, validation writers guide model selection, and test writers remain unseen until final evaluation. The test question then concerns recognition of new writers. It does not merely concern recognition of another image generated by someone already represented in the training set.

D. Lighting and Tracking Interruption

Lighting is primarily a capture-stage variable in this architecture because the CNN receives a rendered canvas rather than the original webcam background. Uneven illumination can reduce tracking quality and introduce missing or unstable points. Once those points have been drawn, the resulting distortion becomes part of the digit image. The study will therefore record lighting conditions and tracking interruptions rather than treating background appearance as a direct CNN input.

A controlled robustness check will compare ordinary indoor lighting with a dimmer but usable condition. Writers will complete the same prompted digit set under both conditions. The study will report the number of capture attempts, unusable traces and valid classified samples separately. This avoids hiding a poor capture rate behind a good classification score computed only on the easiest retained images.

E. Stroke Thickness and Writing Speed

The current painter permits a wide range of brush widths. A fixed width will be selected for the main collection protocol so that fine-tuning is not confounded by arbitrary tool settings. A smaller optional experiment can vary stroke width while keeping the image, split and model constant. The aim is to discover whether shape information survives resizing, especially for digits with narrow gaps or enclosed regions.

Writing speed will be recorded as a coarse participant instruction rather than assumed from an unmeasured label. Normal-speed writing forms the primary condition. A supplementary slow or fast condition may be added if a sufficient number of examples can be collected consistently. Any speed analysis will use actual stroke duration or timestamp records when available. These condition-specific observations will remain distinct from the main writer-independent test score.

F. Touchless Classroom Interaction

A classroom example helps define the intended use without claiming a completed user study. A student could write one digit in front of a webcam and receive immediate visual feedback. An incorrect prediction could be corrected by clearing the canvas and trying again. In this setting, the relevant experience includes how easily the hand is tracked, whether drawing feels stable and how long the prediction takes after a request.

The system is not yet an assessed accessibility product or a replacement for conventional writing tools. Some users may find sustained arm movement tiring. Others may prefer a keyboard or pen. A future usability study should therefore ask about comfort and correction effort rather than assuming that touchless interaction is universally easier. The present research measures technical recognition behavior before making broader usability claims.

IV. SYSTEM ARCHITECTURE

A. Overall Structure

The architecture separates live interaction from offline experimentation. The live path acquires frames, detects landmarks, interprets drawing gestures, renders the canvas, preprocesses a completed digit and performs CNN inference. The proposed offline path saves labelled canvas samples, assigns writer-separated partitions, trains candidate models and evaluates fixed checkpoints. Figure 3 summarizes these paths and the point at which the dataset is created.

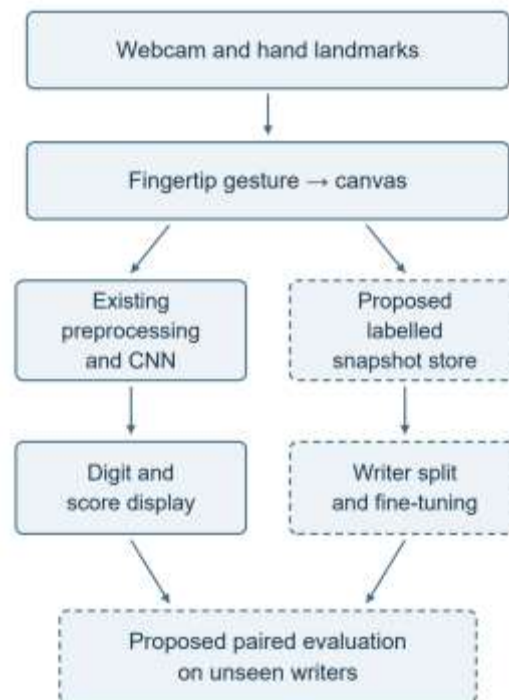


Figure 3: Existing inference path and proposed dataset and fine-tuning extension. Dashed elements indicate proposed work.

This separation allows a drawing to be stored once and evaluated by several models later. Repeating the hand gesture independently for each model would introduce variation in shape and tracking quality. Shared stored samples instead provide a paired comparison. The same immutable test image can be passed through the baseline and the adapted model, and their predictions can be examined against the same label.

B. Capture and Hand-Landmark Layer

OpenCV opens the default webcam and supplies video frames. The painter resizes the working view to 1280 by 720 pixels and mirrors it for screen-based feedback. MediaPipe returns landmarks, and the custom detector converts the normalized positions into pixel coordinates. The index fingertip, landmark 8, provides the principal drawing location. The implementation can detect two hands, although the painting gesture normally uses a single hand.

The detector is shared by the hand-detection demonstration and the painter. This avoids maintaining separate hand-localization logic for each module. The camera and inference loop must still handle failed reads and missing hands. Before research collection begins, the capture path will be checked for graceful cancellation and valid state resets so that missing frames do not silently produce invalid samples.

C. Gesture and Canvas Layer

Finger-state rules use relative landmark positions. Index up with middle down activates drawing; index and middle up activates tool selection. The painter uses a smoothed fingertip position and connects consecutive drawing points. Two detected hands clear the canvas in the current implementation. These rules are implementation choices rather than a learned gesture classifier, and their limitations under rotated hands will be documented.

The canvas is a separate three-channel array with a black background. Colored drawing strokes are stored on that array, while bitwise operations combine the drawing with the live camera frame for display. The recognition input should come from the canvas, not from a screenshot containing the hand, toolbar or background. This boundary is important because the research concerns the digit trace rather than recognition of the entire application window.

D. Recognition and Feedback Layer

The recognition wrapper loads the CNN and a mapping from output classes to digit labels. After preprocessing, inference returns the top class, its score and the scores for all ten digits. Tkinter provides a preview and result window, and Matplotlib displays the confidence graph. A short in-memory history stores recognized digits. Figure 4 shows the recognition display included in the project report.

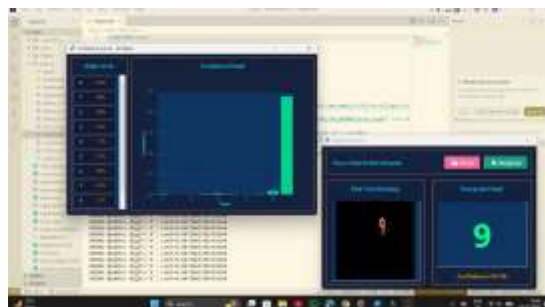


Figure 4: Existing recognition interface showing a digit and class-score graph. A demonstration screenshot is not an accuracy measurement.

An SVM-based fallback also exists in the codebase, but it is not the proposed research baseline. It uses a different input size and training dataset, which would introduce another experimental factor. The primary study will therefore identify the CNN checkpoint explicitly and fail clearly if that model cannot be loaded. A fallback result must not be mixed into a CNN evaluation record without being labelled as a different model.

E. Interface and Research Storage

The desktop launcher and web dashboard provide entry points to the local Python modules. The browser sends a launch request to a lightweight server; recognition itself runs in desktop windows. The current web charts contain demonstration data, so they will not be reused as experimental results. Research figures will be generated from saved evaluation records after experiments are completed.

The proposed storage layer will maintain raw canvas images, normalized images, a sample manifest, split assignments, model checkpoints and prediction records. Each item will have a stable identifier. The application display can remain simple while the research outputs are stored independently. This design supports auditing: an unexpected prediction can be traced back to its source image, preprocessing version, participant code and model version.

Table II: Implementation status and proposed extension.

Component	Present status	Research extension
Hand tracking	Implemented	Log missing or interrupted traces
Virtual painter	Implemented	Add labelled snapshot capture
Digit CNN	MNIST-trained model available	Preserve baseline and adapt a copy
Score display	Implemented	Keep separate from accuracy metrics
Custom dataset	Not collected	Collect and label by writer
Comparison	Not completed	Use fixed held-out images

V. METHODOLOGY AND IMPLEMENTATION

A. Research Design

The proposed study uses a controlled comparison between a fixed MNIST-trained CNN and a version adapted using labelled air-written images. Both models will receive the same held-out inputs through the same preprocessing function. The primary outcome will be macro-F1 across the ten digit classes, with accuracy and per-class results reported alongside it. The experiment is prospective: the following collection targets and training settings are planned choices, not descriptions of completed runs.

The work is organized into collection preparation, dataset construction, baseline evaluation, model adaptation and final analysis. Before collecting data, the team will preserve a copy of the existing model and record the capture settings. The dataset will then be assembled and split by writer. Model development will use only training and validation partitions. Final test predictions will be generated after all preprocessing, training and threshold decisions have been fixed.

B. Preparation of the Capture Interface

The existing painter requires a dedicated collection mode. This mode will prompt the requested digit, accept a completed drawing and save a canvas image with its label. It will also provide an explicit retry option. One sample will correspond to one completed digit, and the canvas will be cleared before the next prompt. Saving a succession of nearly identical video frames from one drawing will not be treated as collecting independent samples.

Collection mode will use a fixed brush color and width for the primary dataset. Tool selection can remain available in the demonstration application, but it will be disabled or logged during research capture. A short practice round will help participants understand the mirror view and drawing gesture. Practice images will be excluded from the evaluation dataset so that the collection record reflects the predefined protocol rather than informal experimentation.

Several implementation checks are needed before collection. The recognition or save action should work after the hand leaves the frame. Stroke coordinates should reset whenever drawing stops, the canvas clears or tracking is lost. Empty canvases should be rejected, and camera failure should end the operation cleanly. These are planned reliability changes; they are not assumed to be fully solved by the current demonstration code.

C. Proposed Dataset Size and Participants

The initial collection target is 20 consenting writers, ten digit classes and 20 completed samples per digit per writer. If achieved, this produces 4,000 original digit images. The target is a feasible starting design,

not a statistically established minimum or a completed dataset. Recruitment and collection time may require adjustment. Any revised target and the final accepted counts will be reported before interpreting experimental findings.

Each writer will complete two sessions with ten samples of each digit per session. Repeated sessions introduce some variation in movement and reduce dependence on a single continuous recording period. Within a session, prompts should be shuffled to avoid producing all examples of one digit in a single repetitive block. The collection software will log the prompt order, but it will not provide model feedback that could encourage writers to imitate a particular classifier's preferred shape.

The principal dataset will use ordinary indoor lighting and a consistent camera position. Participant distance and the allowed drawing area will be explained before capture. A smaller supplementary condition set may examine lower lighting, different speeds or brush widths. These optional samples will carry separate condition labels. They will not silently replace the standard-condition test set or be pooled in a way that changes the meaning of the main comparison.

Table III: Proposed collection and partition targets.

Item	Planned value
Writers	20
Classes	Digits 0–9
Images per digit per writer	20, across two sessions
Total original images	4,000
Training writers and images	12 writers; 2,400 images
Validation writers and images	4 writers; 800 images
Test writers and images	4 writers; 800 images
Actual collection status	Not yet collected

D. Labelling and Quality Review

The prompted digit supplies the intended label, but the saved image still requires review. A blank trace, accidental second digit or clearly unfinished drawing should be marked as an acquisition failure according to a rule written before evaluation. An unusual but interpretable writing style should remain in the dataset. Discarding difficult examples merely because the baseline predicts them incorrectly would bias the comparison in favor of an artificially clean test set.

The manifest will include sample identifier, anonymous writer identifier, session, requested digit, condition, image path and quality status. Capture duration, brush settings and interruption counts can be included when the interface records them reliably. The label will be checked against the prompt and image. Disagreements will be resolved without consulting the candidate models' predictions. Exclusion reasons and the number of rejected attempts will remain available in the dataset summary.

Participants will be assigned codes such as W01 rather than names in the image manifest. Only the drawing canvas is required for classifier training. Storing raw video that contains faces or room details is unnecessary for the main experiment. If additional video is needed to investigate tracking failures, it will

require a separate explanation and participant agreement. Dataset sharing, if later planned, will depend on the permissions obtained during collection.

E. Writer-Separated Partitions

The proposed split allocates 12 writers to training, four to validation and four to testing. Under the 4,000-image target, this corresponds to 2,400 training, 800 validation and 800 test images, with balanced digit counts if collection is complete. Both sessions from a writer remain in the same partition. The split will be created using a recorded random seed or an explicitly stored writer list, consistent with group-aware partitioning [11].

The validation set is used for choosing learning rates, unfreezing depth, preprocessing variants and stopping points. The test set is used only after these choices are frozen. Images derived by augmentation inherit the partition of their original sample and are generated only from training images. Near-duplicates must be removed or grouped before splitting. Figure 5 illustrates the proposed separation and the absence of test feedback during model development.

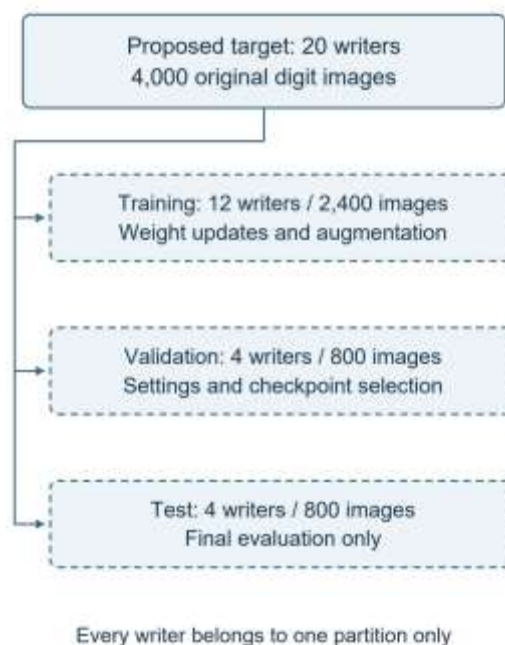


Figure 5: Proposed writer-separated split for the 4,000-image collection target. Counts are planned, not collected.

Four test writers provide only a limited view of writer diversity. The study will therefore report writer-specific results instead of relying solely on a pooled score. If enough participants are available, a supplementary grouped cross-validation analysis may test sensitivity to writer assignment. Its protocol must be fixed independently of the held-out test decision. A large number of samples from very few writers is not equivalent to a large number of independent writers.

F. Image Preprocessing

The research preprocessing pipeline will first operate on the stored canvas and identify non-background pixels. It will crop the union of meaningful stroke components with a small margin. The crop will be padded to a square without changing its aspect ratio, resized to fit within a 20 by 20 region and placed in a 28 by 28 image. Centering based on foreground mass can then reduce translation variation. Pixel values will be scaled consistently for the CNN.

Foreground polarity must be checked explicitly. The intended CNN representation is a bright digit on a dark background. Because the painter already uses a dark canvas, unconditional inversion can convert the background into the apparent foreground. The preprocessing code will be inspected using blank, thin-stroke and thick-stroke inputs before use. This is particularly important because the existing implementation combines thresholding operations whose behavior can differ across stroke colors.

Morphological cleanup will be kept small enough to preserve loops and gaps. Aggressive closing may change a three into an eight-like shape; excessive erosion may break a thin one into disconnected fragments. The proposal therefore treats preprocessing as a controlled factor. A raw-to-normalized preview will be stored for inspection, and the exact transformation will be applied to both baseline and fine-tuned models in the primary comparison.

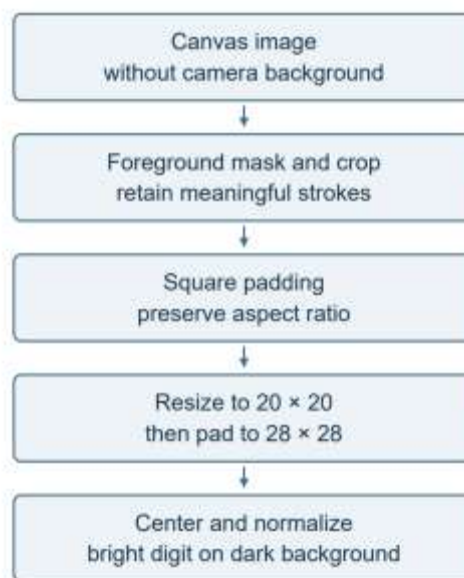


Figure 6: Proposed preprocessing sequence for a stored canvas image. The diagram specifies operations and contains no experimental output.

An initial baseline with the original application preprocessing will be recorded separately from the baseline with corrected common preprocessing. This distinguishes an engineering improvement from a learned improvement. If preprocessing alone produces most of the gain, the results should state that clearly. Fine-tuning should receive credit only for the additional difference observed when both models share the same preprocessing and evaluation samples.

G. Existing CNN and Source Training

The training definition in the current code accepts a 28 by 28 single-channel image. It contains two convolution layers with 64 filters, pooling and dropout, followed by two convolution layers with 128 filters, another pooling stage and dropout. Flattened features enter dense layers of 256 and 128 units, followed by ten softmax outputs. This architecture will be verified against the saved checkpoint before experiments rather than inferred solely from the training script.

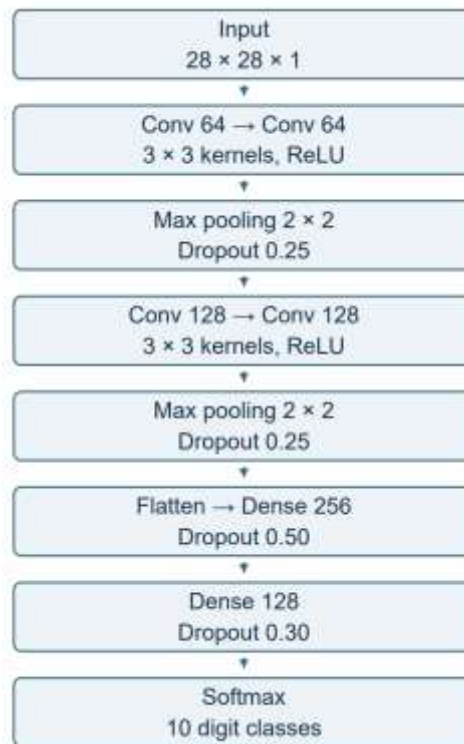


Figure 7: CNN structure defined in the existing training code. Dropout is used during training; the output has ten digit classes.

The source training code specifies Adam, categorical cross-entropy, batch size 128 and 15 epochs on MNIST. These are code settings, not a recovered training log or a measured accuracy claim. The present implementation passes the MNIST test partition as validation data during training. For a clean reproducible rerun, a validation subset should instead be drawn from the MNIST training partition, leaving the official test partition for final source-domain evaluation.

The saved baseline will be preserved without further updates. Its model hash, class mapping, package versions and expected input normalization will be recorded. Fine-tuning must begin from a copy of that checkpoint. The general SVM training utility currently removes existing model files, so it should not be used as the fine-tuning entry point. A separate research training script will maintain baseline and adapted checkpoints under distinct names.

H. Proposed Fine-Tuning Procedure

The initial adaptation stage will freeze the convolution blocks and update the dense classification layers using target training images. A provisional learning rate of 0.0001 and batch size of 32 will be used as a starting configuration. A maximum of ten epochs with validation-based early stopping is proposed. These values will be treated as development settings, with any changes documented rather than selected through repeated inspection of test accuracy.

In the second stage, the upper convolution block will be unfrozen and the model recompiled. A smaller provisional learning rate of 0.00001 will limit the size of updates to the pretrained representation. The planned upper limit is 20 epochs, with early stopping after five epochs without validation improvement and restoration of the best validation checkpoint. Lower layers will initially remain frozen. Unfreezing additional layers is an optional validation-selected variant, not a mandatory route to improvement [10].

The ten output classes already match the target task, so the final layer does not need to be expanded. The study will retain the label mapping from zero through nine and verify that evaluation code uses the same order. Class weights will be unnecessary if the training set is balanced; if actual collection counts become unequal, any weighting strategy will be decided using training counts alone. The chosen loss and sampling policy will be included in the experiment record.

I. Data Augmentation and Controls

Controlled augmentation will introduce small rotations, translations and scale changes to training images. Initial candidate limits are approximately ten degrees of rotation, ten percent translation and ten percent scale variation. These bounds are proposed and must be reviewed visually because some transformations can change digit identity. Horizontal flips, vertical flips and large rotations will be excluded from the main protocol. Augmentation will not be applied to validation or test images during ordinary evaluation.

The main conditions are B0, the unchanged CNN with original preprocessing; B1, the unchanged CNN with standardized research preprocessing; and F1, the fine-tuned CNN with that same standardized preprocessing. Comparing B0 with B1 estimates the preprocessing effect, while comparing B1 with F1 estimates the adaptation effect. A model trained from scratch on target training data, S1, is an optional control for the value of MNIST initialization. All conditions will use the same label set and test samples.

Table IV: Proposed experiment conditions.

ID	Training	Input processing
B0	Existing MNIST checkpoint	Original application pipeline
B1	Same checkpoint as B0	Standardized research pipeline
F1	MNIST initialization plus target fine-tuning	Same pipeline as B1
S1	Target-only training from random initialization; optional	Same pipeline as B1

If resources permit, training conditions will be repeated with three recorded seeds while keeping the main split fixed. The report will include the mean and spread across runs rather than selecting only the best run. A no-augmentation fine-tuning variant can isolate the effect of augmentation. These supplementary comparisons will be limited to questions the dataset can support, avoiding a large search whose apparent winner is determined by chance.

J. Evaluation and Reproducibility

Every test prediction will be saved with sample ID, ground-truth digit, predicted digit, class-score vector, model ID and timing information. The stored record supports a confusion matrix and per-writer analysis without rerunning live drawing. The baseline and adapted models will be compared on paired examples. Error inspection will show both the original canvas and normalized image so that failures can be assigned to capture, preprocessing, ambiguity or classification with appropriate caution.

The experiment package will include a dataset manifest, a fixed split file, training configuration, checkpoint identifiers, evaluation script and generated result tables. Training and inference environments will record Python, TensorFlow, Keras, OpenCV and MediaPipe versions. Raw data will be kept unchanged, while normalized versions will carry a preprocessing-version identifier. This prevents an unnoticed software change from producing incomparable results under the same experiment name.

K. Runtime Measurement

Interactive performance will be measured separately from offline classification. Per-frame tracking time, preprocessing time and CNN inference time will be recorded using a monotonic timer. End-to-end recognition latency will begin at a valid recognition request and end when the result is ready for display. Human writing time will be reported separately because it is not model inference time. Initial model loading will also be separated from warmed-up inference.

The proposed report will state the actual processor, memory, camera settings and compute device used. Median latency and a high percentile will be more informative than a single best timing. Offline batch throughput will not be presented as live frame rate. Fine-tuning changes weights without necessarily changing the network architecture, so improved accuracy need not imply faster inference. Both claims require their own measurements.

VI. THEORETICAL ANALYSIS

A. Source and Target Distributions

Let the source dataset contain conventional handwritten digit images and let the target dataset contain images rendered from fingertip trajectories. Both use the label set $Y = \{0, 1, \dots, 9\}$. The proposed problem assumes that the source and target input distributions may differ while the required output meaning remains fixed. Fine-tuning attempts to adapt parameters learned from the source task using labelled target examples. It does not guarantee that the two distributions become identical.

The initial parameter set is denoted by θ_S . Adaptation starts from θ_S and seeks parameters θ_T that reduce classification loss on the target training partition. The held-out target partition estimates whether that change generalizes. A decrease in training loss alone is insufficient evidence because it may reflect memorization. The validation and test boundaries are therefore part of the theoretical logic of the study, not merely a file-management convenience.

B. Trajectory Smoothing and Rendering

The painter smooths fingertip coordinates before connecting successive positions. If p_t is the detected position and s_t is the displayed position at time t , exponential smoothing can be expressed as the weighted combination below. The existing painter uses $\alpha = 0.5$. Smaller α values suppress rapid variation more strongly but also delay the displayed response to genuine movement.

$$s_t = (1 - \alpha)s_{t-1} + \alpha p_t \quad (1)$$

This trade-off matters because smoothing can alter the geometry of corners and short strokes. A stable line may look better while losing a distinguishing detail. The collection protocol will keep the smoothing setting fixed for the main comparison. If smoothing is varied later, that experiment will be reported as a capture-policy analysis rather than attributed to CNN adaptation. Stroke continuity must also be reset correctly when the gesture leaves drawing mode.

C. Classification and Learning Objective

For each processed image, the CNN produces ten scores through a softmax output. The predicted digit is the class with the highest score. Cross-entropy penalizes assigning low probability to the correct class. During fine-tuning, gradient updates modify only the layers whose parameters are marked trainable. Keeping lower layers fixed constrains the amount of adaptation when target data are limited.

$$L = -(1/N) \sum_i \sum_c y_{ic} \log(p_{ic}) \quad (2)$$

In Equation 2, N is the number of training examples, y_{ic} is the indicator for the correct digit class and p_{ic} is the predicted class probability. The objective is optimized on training data. Validation loss guides

stopping, and test metrics estimate generalization. The same loss value cannot directly explain whether an error arose from a distorted trace, an ambiguous writing style or poor feature adaptation; saved example inspection remains necessary.

D. Classification Metrics

Accuracy is the proportion of correctly predicted test images. For a given digit class c , precision measures the fraction of predictions of c that are correct, while recall measures the fraction of actual c examples that are recovered. F1 combines precision and recall. Macro-F1 gives every digit equal weight, which is useful if the final dataset is slightly unbalanced or one difficult class performs much worse than the others [12].

$$\text{Accuracy} = \text{correct predictions} / \text{test samples} \quad (3)$$

$$\text{Precision}_c = TP_c / (TP_c + FP_c) \quad (4)$$

$$\text{Recall}_c = TP_c / (TP_c + FN_c) \quad (5)$$

$$F1_c = 2 P_c R_c / (P_c + R_c) \quad (6)$$

$$\text{Macro-F1} = (1/10) \sum_c F1_c \quad (7)$$

TP, FP and FN denote true positives, false positives and false negatives under a one-class-versus-rest interpretation. In Equation 6, P_c and R_c denote precision and recall for class c . If a denominator is zero, the evaluation software will use a documented zero-division convention. Class support will be shown so that metrics based on few samples are not overinterpreted. A confusion matrix will place true labels on rows and predicted labels on columns, with both counts and row-normalized values available.

E. Improvement and Uncertainty

The primary improvement is the difference between fine-tuned and baseline macro-F1 on the same target test set. Accuracy improvement will also be stated as an absolute percentage-point difference. Relative percentage changes can be misleading when baseline accuracy is already high, so they will not replace the underlying scores. Both successful and failed predictions will remain part of the paired comparison.

$$\Delta F1 = \text{Macro-F1}(F1) - \text{Macro-F1}(B1) \quad (8)$$

Samples from one writer are related, so treating every image as an independent participant can exaggerate statistical certainty. The analysis will show results for each held-out writer and variation across training seeds. A supplementary writer-level resampling analysis may be reported with caution, but four test writers provide limited evidence for a precise population interval. Broader claims would require additional independent writers and collection conditions.

F. Confidence and Rejection

The maximum softmax score expresses the model's relative preference among its known classes. It is not automatically a calibrated probability that the answer is correct [13]. The existing application accepts very low scores, so displaying a digit does not establish that the prediction is reliable. The research will first evaluate unconditional ten-class predictions, then consider a separate rejection policy selected on validation data.

If a confidence threshold is used, the report will state both accuracy among accepted predictions and coverage, the fraction of samples receiving a prediction. A system that rejects most difficult inputs may appear accurate while being inconvenient to use. Empty-canvas detection must occur before this threshold because a blank image is not one of the ten digit classes. The proposed evaluation will count invalid captures separately and describe how they affect end-to-end task completion.

G. Computational Considerations

The capture pipeline processes a full camera frame, while the digit CNN processes only a small normalized image. Tracking, display refresh and classification therefore have different costs. Increasing camera

resolution may affect tracking and rendering without changing the CNN input size. Likewise, additional stored dataset images increase training effort but do not necessarily increase the work required for one prediction.

The planned adaptation retains the same CNN topology for the primary comparison. This makes model size and inference operations approximately comparable, although actual runtime can still vary with software and hardware. If a later experiment replaces the classifier with a smaller network, its speed and accuracy should be treated as a new model-design comparison. Weight adaptation and architecture optimization answer different questions.

VII. RESULTS AND EXPECTED OUTCOMES

A. Present Implementation Evidence

The existing project contains a working design for hand tracking, gesture-controlled drawing and digit prediction, supported by source code and screenshots in the project report. The report shows hand landmarks, colored canvas strokes and a recognition window displaying the digit nine with a class-score graph. These materials demonstrate the intended interaction and the availability of the relevant components. They do not provide a controlled test population or a dataset-wide performance estimate.

The students' completed learning work uses MNIST. No custom air-writing fine-tuning run, writer-separated test evaluation or experimental comparison between baseline and adapted models has yet been completed. Consequently, this section reports implementation evidence and expected research outcomes rather than numerical gains. Accuracy, F1-score, confusion-matrix values and latency measurements will be obtained through the protocol described above; no estimated values are substituted for those measurements.

B. Expected Adaptation Behavior

The main hypothesis is that labelled air-written training images will help the CNN recognize shapes produced by the gesture interface. Improvement may occur if the source model already contains useful digit features but is sensitive to target-specific stroke patterns. The result may be small if standardized preprocessing resolves most of the mismatch. Fine-tuning may also reduce performance if the collected dataset is too narrow, noisy or biased toward the training writers.

The expected research output is therefore a measured comparison, not a predetermined accuracy target. A supported positive result requires stronger held-out performance under the same preprocessing and test set. If the fine-tuned model improves only for certain writers or digit classes, that limitation will be reported. A negative or inconclusive result still helps identify whether future effort should focus on dataset diversity, capture stability or model design.

C. Planned Result Presentation

The primary result table will contain model condition, number of evaluated images, accuracy, macro-precision, macro-recall and macro-F1. A second table will show per-digit precision, recall, F1 and support. Baseline and adapted confusion matrices will use identical axes and color scales. A class-wise recall chart will reveal whether an overall improvement is broadly distributed or driven by a small subset of digits.

Training curves will plot actual training and validation loss for each completed epoch. They will be generated from the saved history rather than drawn to illustrate an expected trend. A paired error gallery will show cases corrected by fine-tuning, cases newly misclassified and errors shared by both models. Any confidence values displayed beside examples will be identified as model outputs for those images, not general reliability estimates.

Table V: Evidence available now and outputs planned.

Item	Current evidence or planned output
Interface	Existing report screenshots and implementation
MNIST training	Completed by the student team; no benchmark score asserted here
Target dataset	Proposed collection; no accepted sample count yet
Fine-tuning	Proposed; no adapted checkpoint evaluated
Recognition metrics	To be measured on writer-separated test images
Runtime	To be measured on the documented test hardware

D. Capture and Runtime Outcomes

The capture report will count attempted drawings, accepted digit images and rejected acquisitions for each condition. It will also summarize the reasons for rejection. This provides context for the classifier metrics: an accurate model cannot compensate for an interface that frequently fails to produce a usable image. The denominator used for each rate will be stated explicitly so that acquisition success and classification success remain distinguishable.

Runtime results will include model loading time, warmed-up inference latency and end-to-end response after recognition is requested. Frame rate will be measured on the tested device rather than copied from the introductory report. If the adapted model uses the same architecture, similar inference cost is a reasonable expectation, but it remains a hypothesis until measured. A small accuracy gain accompanied by unstable interface response may have limited practical value.

E. Interpretation of Possible Outcomes

If B1 substantially exceeds B0 while F1 only slightly exceeds B1, input standardization would be the main contributor. If F1 improves over B1 across most test writers, the evidence would support supervised adaptation within the evaluated setting. If S1 performs similarly to F1, source initialization may offer little final-score benefit at the chosen dataset size, although training speed or run-to-run stability could still differ.

If performance varies sharply across writers, the study should investigate whether training data cover the relevant styles. If errors cluster under dim lighting, tracking quality and acquisition failures should be examined before changing the classifier. Interpretation will follow these competing explanations instead of assuming that every improvement comes from a more capable CNN. This approach makes the eventual findings useful for improving the complete application.

VIII. LIMITATIONS

A. Current Evidence and Sample Size

The research extension is currently at the proposal stage. The project report and screenshots document the prototype, but they do not establish its generalization to unseen writers. The planned 20-writer dataset is a manageable starting point and may still be too small to represent the variety of hand movements and digit styles encountered in wider deployment. A test involving four writers should therefore be interpreted as a local study.

The absence of completed adaptation experiments limits the conclusions that can be drawn now. No superiority claim is made for the proposed fine-tuned model. Even after collection, a narrow participant group or a single device may constrain external validity. Reporting the actual sample composition and capture setup will help readers judge which settings the eventual results support.

B. Tracking and Gesture Constraints

Finger-state comparisons work best when the hand is clearly visible and roughly aligned with the camera. Occlusion, rotation, fast movement and poor lighting may cause unstable landmarks. The two-hand clear gesture can also erase a drawing if another hand enters the view. Research collection will need explicit instructions and controlled state handling to prevent accidental loss of samples.

The prototype's stroke smoothing reduces jitter but can soften sharp corners. Fixed camera dimensions and tool-selection regions may not suit every screen or user. These interface choices affect the images later used for learning. They cannot be removed from the scientific interpretation by considering only the final CNN score, especially when an error originated during capture.

C. Representation and Vocabulary

The primary classifier receives a static rendering of a complete trajectory. This representation discards stroke order, timing and pauses unless those data are saved separately. Two different movements can therefore produce similar images, while missing points can create a misleading shape. Sequence-based models may exploit information that this image approach ignores, but they require additional data and a different evaluation design.

The vocabulary contains ten isolated digits. Letters, mathematical symbols, connected numbers and words are outside the current experiment. Generalization to these tasks would require new labels, segmentation decisions and suitable training data. A title referring to handwriting recognition should not be interpreted as evidence that unrestricted handwriting recognition has been implemented.

D. Dataset and Model Bias

Participants may adapt their drawing style after watching examples, which can reduce natural variation. Prompt order, repeated practice and collection fatigue may also influence traces. The protocol will limit these effects through practice separation, shuffled prompts and session breaks, but it cannot eliminate every source of bias. Keeping the raw images and collection record makes these influences easier to investigate later.

Fine-tuning may overfit target writers or reduce source-domain performance. Validation-based stopping and augmentation help manage these risks but do not guarantee reliable transfer. A separate source-domain evaluation will be useful for identifying forgetting. Such a result should be reported as a trade-off rather than hidden because the main target score improves.

E. Application and Accessibility Boundaries

The current system is a local prototype with a web launcher, not a deployed cloud service. Browser login and dashboard statistics are demonstration features rather than a research account-management or telemetry system. Their appearance does not establish secure multi-user access. The research dataset and model-evaluation pipeline will remain separate from those interface elements.

Touchless input may be useful in some contexts, but sustained arm movement can cause fatigue and may exclude users with limited upper-limb mobility. No clinical or accessibility trial has been completed. Educational and assistive uses are therefore possible application directions, not validated outcomes. A later user study should include participant feedback and alternative input options.

IX. FUTURE WORKS

A. Larger and More Varied Data Collection

After completing the initial protocol, the dataset can be extended with more independent writers, camera devices and capture environments. The main priority should be diversity of contributors rather than repeated near-identical samples. A public benchmark would require a stable label scheme, a documented split, participant permission and a clear data-use licence. Release should preserve privacy by distributing rendered digit images whenever raw video is unnecessary.

Additional collection rounds can target known failure categories identified by the first study. For example, if open-top fours are consistently misclassified, targeted samples can test whether the issue is representation or insufficient training coverage. Those new samples must enter a new development cycle with an untouched evaluation set. Repeatedly adding test errors to training would otherwise make the benchmark unsuitable for measuring progress.

B. Temporal Recognition and Better Delimiting

Saving timestamped fingertip coordinates would permit experiments with sequence models that retain stroke order and duration. Such models could be compared with raster-image CNNs on the same writing attempts. They may also help distinguish accidental transitions from intended strokes. However, they introduce new alignment, sampling and sequence-length choices that need their own controlled evaluation. The explicit recognition button can later be supplemented with a stable completion gesture or inactivity-based trigger. Completion detection must be evaluated separately because an early trigger can truncate a digit and a late trigger can delay feedback. Multi-digit recognition would also require segmentation or a sequence-level output model. These extensions should follow successful isolated-digit evaluation rather than being assumed from the current interface.

C. Deployment and User Experience

Model compression and smaller networks could be explored for lower-resource computers or mobile devices. Their benefit should be measured through model size, energy or latency together with recognition quality. A browser-native version would require a different camera and inference integration from the existing local launch server. It would be a deployment extension, not a property of the current web dashboard.

A user study could assess ease of learning, fatigue, correction effort and preferred feedback. Predictions with uncertain scores might offer a retry prompt or a short candidate list. Such changes should be tested for their effect on successful task completion rather than only visual appeal. User-centered evidence would complement, but not replace, the classifier comparison proposed in this paper.

D. Visual Plan for Dataset Development

Figures 8 and 9 translate the proposed collection protocol into allocation and class-balance targets. The pie chart assigns 60% of the 4,000 original images to training and 20% each to validation and testing. The stacked bars show the resulting target of 240 training, 80 validation and 80 test images for every digit. These values describe a balanced collection plan; they are not observed dataset statistics.



Figure 8: Proposed dataset allocation. Percentages refer to image counts, with writers separated between partitions.



Figure 9: Planned class balance across digits 0–9. Each stacked bar totals 400 original images; augmentation is excluded.

E. Collection Scale and Training Budgets

Figure 10 considers optional expansion from 20 to 30 or 40 writers while retaining 200 accepted original images per writer. The totals follow from that assumption and do not predict accuracy. Figure 11 summarizes the two proposed fine-tuning stages. Epoch limits are training budgets rather than completed runs; validation-based early stopping can reduce the actual number.

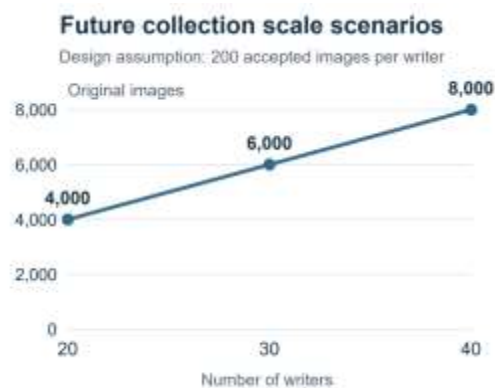


Figure 10: Illustrative collection-scale scenarios. Twenty writers is the initial proposal; 30 and 40 writers are optional future scenarios.

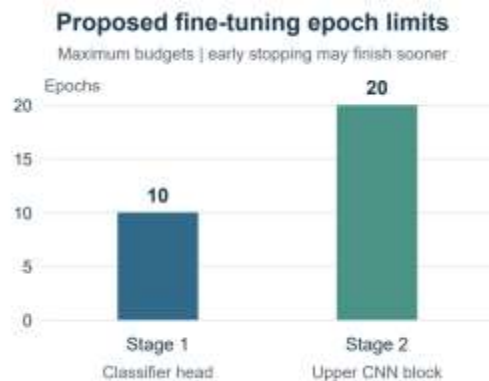


Figure 11: Proposed maximum epoch budgets of 10 and 20 for the two fine-tuning stages. No training curve or performance gain is implied.

F. Implementation Schedule and Feature Roadmap

The eight-week schedule in Figure 12 is an illustrative sequence that can be revised after the capture pilot and participant availability are known. Dataset quality checks and a fixed writer split precede model comparisons. Figure 13 places temporal recognition, deployment and user studies after the initial evaluation; these are optional research directions requiring their own validation.

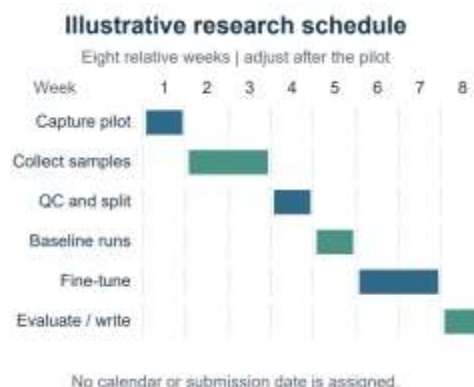


Figure 12: Tentative research schedule in relative weeks. The schedule is a planning aid, with no assigned submission date.



Figure 13: Proposed dependencies between the initial study and later feature updates. Branches indicate possible extensions, not completed functionality.

X. CONCLUSION

This work connects an existing gesture-based virtual painter with a proposed investigation of air-written digit recognition. The implemented project supplies webcam capture, hand landmarks, canvas rendering and an MNIST-based digit-recognition interface. The research extension will collect labelled air-written samples and adapt the CNN to the characteristics of those images. Its central question is whether this adaptation improves recognition for writers excluded from training.

The proposed methodology defines a collection target, writer-separated partitions, consistent preprocessing, preserved baseline checkpoints and a staged fine-tuning procedure. It separates the effects of input normalization from the effects of learning, and it treats confidence, capture success and latency as distinct quantities. These decisions make the planned comparison more informative than a demonstration based on a few selected predictions.

Custom dataset collection and fine-tuning remain to be completed. The expected contribution is a reproducible experiment and documented evidence about when adaptation helps the current system. Any final claim of improvement will depend on measured held-out results. The study provides a practical route from an undergraduate prototype to a focused research investigation without assuming that an MNIST score represents real-world air-writing performance.

ACKNOWLEDGEMENT

The student authors thank Dr. Sonia Sharma for academic guidance and the Department of Computer Science and Engineering at Malout Institute of Management and Information Technology for the educational setting in which the project was developed. They also acknowledge the researchers and open-source communities responsible for MNIST, MediaPipe, OpenCV, TensorFlow, Keras and the evaluation tools used in the proposed methodology.

REFERENCES

1. Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. Online source
2. Keras, "MNIST digits classification dataset," Keras API documentation. Online source
3. F. Zhang et al., "MediaPipe Hands: On-device Real-time Hand Tracking," arXiv:2006.10214, 2020. Online source
4. Google AI Edge, "Hand landmarks detection guide for Python," MediaPipe documentation. Online source
5. S. Mukherjee, A. Ahmed, D. P. Dogra, S. Kar and P. P. Roy, "Fingertip Detection and Tracking for Recognition of Air-Writing in Videos," *Expert Systems with Applications*, vol. 136, pp. 217–229, 2019. doi: 10.1016/j.eswa.2019.06.034. Online source
6. M. Wu, K. Huang, Y. Cai, S. Hu, Y. Zhao and W. Wang, "Finger in Camera Speaks Everything: Unconstrained Air-Writing for Real-World," arXiv:2412.19537, 2024. Online source
7. S. J. Pan and Q. Yang, "A Survey on Transfer Learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010. doi: 10.1109/TKDE.2009.191. Online source
8. M. Wang and W. Deng, "Deep Visual Domain Adaptation: A Survey," arXiv:1802.03601, 2018. Online source
9. Y. Ganin et al., "Domain-Adversarial Training of Neural Networks," *Journal of Machine Learning Research*, vol. 17, no. 59, pp. 1–35, 2016. Online source

10. F. Chollet, “Transfer learning & fine-tuning,” Keras developer guide, 2020, updated 2023. Online source
11. Scikit-learn developers, “GroupShuffleSplit,” API documentation. Online source
12. Scikit-learn developers, “*classification_report*,” API documentation. Online source
13. C. Guo, G. Pleiss, Y. Sun and K. Q. Weinberger, “On Calibration of Modern Neural Networks,” Proceedings of the 34th International Conference on Machine Learning, PMLR, vol. 70, pp. 1321–1330, 2017. Online source
14. A. Qedear, A. AlMatrafi, A. Al-Sowat, A. Saigh and A. Alayed, “Real-Time Air-Writing Recognition for Arabic Letters Using Deep Learning,” Sensors, vol. 24, no. 18, art. 6098, 2024. doi: 10.3390/s24186098. Online source
15. B. Feng, G. Chen, H. Chen and A. Wang, “RingRT: recognizing air-writing on the palm through an IMU-integrated ring,” CCF Transactions on Pervasive Computing and Interaction, vol. 8, pp. 30–43, 2026; first published online 30 October 2025. doi: 10.1007/s42486-025-00209-9. Online source
16. Y. Nakamura and H. Miwa, “Mobile-AeroText: Air Writing Recognition of Japanese Hiragana for Practical Touchless Text Input,” Journal of Information Processing, vol. 34, pp. 75–83, 2026. doi: 10.2197/ipsjip.34.75. Online source
17. F. Imbert, S. Corbillé, H. Han and E. H. Barney Smith, “Domain Adaptation Based Pipeline for Character Classification and Handwritten Text Recognition,” International Journal on Document Analysis and Recognition, published online 30 March 2026. doi: 10.1007/s10032-026-00575-9. Online source