International Journal of Computer Applications (0975 - 8887)
Volume 187 - No.47, October 2025

An MLP Baseline for Handwriting Recognition using Planar Curvature and Gradient Orientation

Azam Nouri
Assistant Professor
Department of Science, Technology & Mathematics
Lincoln University, USA

ABSTRACT

This study investigates whether second-order geometric cues—planar curvature magnitude, curvature sign, and gradient orientation—are sufficient on their own to drive a multilayer perceptron (MLP) classifier for handwritten character recognition (HCR), offering an interpretable alternative to convolutional neural networks (CNNs). Using these three handcrafted feature maps as inputs, the curvature–orientation MLP achieves **97 %** accuracy on MNIST digits and **89 %** on EMNIST letters. These results underscore the discriminative power of curvature-based representations for handwritten character images and demonstrate that competitive performance is achievable with lightweight, explicitly engineered features.

Keywords

Handwritten recognition; planar curvature; gradient orientation; multilayer perceptron; MNIST; EMNIST

1. INTRODUCTION

Handwritten-character recognition (HCR) is commonly approached with convolutional neural networks (CNNs), which learn features directly from pixels. However, much of the discriminative information lies in the *geometry of strokes*—where they bend, how sharply they bend, and on which side of the tangent the curve turns. This paper investigate *planar curvature* as an explicit, interpretable descriptor for HCR and examines how far a simple multilayer perceptron (MLP) can go when fed only with curvature-derived maps.

1.1 Motivation: Why Curvature?

Curvature magnitude highlights loops, hooks, and corners; curvature sign distinguishes concave from convex turns; and local gradient orientation provides first-order context around each bend. Together, these channels surface the cues humans rely on to tell apart visually similar letters (e.g., C vs. G).
Curvature-based maps reveal exactly what distinguishes many handwritten glyphs: where a stroke bends, how much it bends, and which way it bends.

1.2 Approach and Findings

Three maps per glyph are computed—curvature magnitude $|\kappa|$, curvature sign $\operatorname{sign}\kappa$, and gradient orientation θ—which are stacked and flattened into a 2352-D vector, then used to train a compact MLP classifier. On standard benchmarks, the model reaches **97 %** test accuracy on MNIST and **89 %** on EMNIST Letters.

1.3 Contributions

A practical, self-contained pipeline for estimating discrete planar curvature and orientation on raster glyphs, with simple stabilization and normalization choices.

A lightweight MLP baseline driven solely by curvature-derived maps, with strong results on MNIST and EMNIST Letters.

Reproducible code and training protocol to facilitate follow-up work (see Code Availability).

2. RELATED WORK

Contour-based gradient features has appeared in HCR pipelines since the 1990s [9, 7]. Recent CNN variants incorporate learnable higher-order derivative filters for edge-aware feature extraction [12, 13], leveraging convolutional inductive biases.
The present work differs by coupling curvature to an all-dense MLP, thereby maintaining a fixed memory footprint and explicit interpretability [5].

3. BACKGROUND: PLANAR CURVATURE

3.1 Geometric Intuition

Imagine tracing a pen along a stroke. Where the pen goes straight, the turning rate is near zero; where it bends sharply (e.g., around a loop or at a hook), the turning rate spikes. The *magnitude* of that turning rate indicates how much the stroke bends; its *sign* indicates which way it bends (concave vs. convex relative to the local tangent).

3.2 Analytical Definition (Continuous Curves)

For a twice-differentiable planar curve $\gamma(t) = (x(t), y(t))$, the signed curvature is [3]

$$\kappa(t) = \frac{x'(t)\,y''(t) - y'(t)\,x''(t)}{\left(x'(t)^2 + y'(t)^2\right)^{3/2}}, \tag{1}$$

with the conventional right-handed sign (counterclockwise positive).

3.3 Normalized Curvature Function and Invariances

Reparameterize by arc length $s \in [0,1]$ and define $k(s) = L\,\kappa(t(s))$ with $L = \int_0^1 \|\gamma'(t)\|\,dt$. Two curves are similar (up to uniform scaling, rotation, and translation) if and only if they share the same $k(s)$ [11].

3.4 Practical Pitfalls and Remedies

Discrete curvature can spike at endpoints and junctions and be noisy under weak gradients; gentle stabilization and reflection at borders mitigate these effects.

4. METHODOLOGY

This section specifies (i) how curvature and orientation maps are computed on raster glyphs and (ii) the MLP classifier and training protocol used for all experiments. Only second-order geometry (planar curvature) and first-order orientation are used—no other edge features enter the descriptor.

4.1 Preprocessing

Input images are 28×28 grayscale glyphs with intensities in $[0,1]$. No deskewing, thinning, or binarization is performed.

4.2 Scale-Space Derivatives

Spatial derivatives are computed using 3×3 Sobel operators to obtain first-order gradients I_x, I_y and second-order terms I_{xx}, I_{yy}, I_{xy}. Zero padding is used at the image boundary.

4.3 Curvature and Orientation Maps

Let $g_x = I_x$ and $g_y = I_y$. The signed image curvature κ at each pixel is computed as

$$\kappa \;=\; \frac{I_{xx}\,g_y^2 \;-\; 2\,I_{xy}\,g_x g_y \;+\; I_{yy}\,g_x^2}{\left(g_x^2 + g_y^2 + \varepsilon\right)^{3/2}}, \tag{2}$$

with $\varepsilon = 10^{-8}$ to avoid division by zero when $\|\nabla I\|$ is small [2]. The gradient orientation is $\theta = \operatorname{atan2}(g_y, g_x) \in (-\pi, \pi]$, mapped to $[0,1]$ as $\theta = (\theta + \pi)/(2\pi)$. Three channels are retained: (i) curvature magnitude $|\kappa|$; (ii) curvature sign $\operatorname{sign}(\kappa) \in \{-1, 0, 1\}$; and (iii) orientation θ.

4.4 Vectorization

The three 28×28 maps are stacked along the channel dimension and flattened to a single 2352-D feature vector per glyph.

4.5 Classifier Architecture

A compact MLP processes the 2352-D vector using fully connected (FC) blocks with batch normalization (BN), ReLU, and dropout. The configuration is summarized in Table 1. For digit recognition, the output layer has 10 units; for letters, 26 units.

4.6 Training Protocol

Weights are randomly initialized (Glorot uniform). Optimization uses Adam (learning rate 10^{-3}), batch size 128, and sparse cross-entropy. Early stopping monitors validation accuracy with patience 8 and restores the best epoch. A `ReduceLROnPlateau` scheduler halves the learning rate on validation-loss plateaus (patience 4, floor 10^{-6}). No L_2 penalty or augmentation is used.

Table 1. MLP for curvature–orientation inputs.

Layer	Dimension	Components
Input	2352	–
Hidden 1	2048	FC + BN + ReLU + Dropout (0.5)
Hidden 2	1024	FC + BN + ReLU + Dropout (0.5)
Hidden 3	512	FC + BN + ReLU + Dropout (0.4)
Hidden 4	256	FC + BN + ReLU + Dropout (0.3)
Output (digits)	10	FC + Softmax
Output (letters)	26	FC + Softmax

Table 2. Stratified 80/20 split of TFDS train+test (with 10% of the training portion used for validation at fit time).

Dataset	Total	Train	Val	Test
MNIST (0–9)	70,000	50,400	5,600	14,000
EMNIST Letters (A–Z)	145,600	104,832	11,648	29,120

4.7 Reproducibility

Experiments are implemented in TensorFlow 2.x/Keras with fixed seeds for NumPy and TensorFlow. Unless noted, results report the held-out test split after a single run with early stopping. Code and scripts are available (Code Availability).

5. EXPERIMENTAL SETUP

5.1 Datasets

MNIST digits [4] (70,000 images) and EMNIST Letters [1] (145,600 uppercase letters) are used. Data are loaded via TensorFlow Datasets (TFDS) [10].

5.2 Split Protocol

The TFDS `train` and `test` partitions are concatenated, followed by a stratified 80/20 split. During training, Keras holds out **10% of the training portion** as validation via `validation_split=0.1`. Table 2 shows the resulting sizes.

5.3 Label Handling

For the EMNIST Letters dataset, which consists of 26 classes, the implementation shifts the labels to the range 0–25 for consistency with zero-based indexing.

5.4 Evaluation Metrics

Both datasets use balanced classes with stratification. Accordingly, top-1 accuracy is the primary metric. Macro-averaged precision/recall/F1 and micro-averaged scores are also reported.

6. RESULTS

6.1 Aggregate Metrics

Table 3 reports macro/micro F1, precision and recall. On MNIST, the model reaches **97%** accuracy. On EMNIST Letters, accuracy is **89%**.

Table 3. Aggregate metrics on the held-out test sets (present run, whole-percentage rounding).

Dataset	**Accuracy**	**Precision**	**Recall**	**F1$_{macro}$**	**F1$_{micro}$**
MNIST	**97%**	97%	97%	97%	97%
EMNIST Letters	**89%**	89%	89%	89%	89%

6.2 Confusion-Matrix Analysis (MNIST)

Figure 1 shows the confusion matrix for MNIST digits. The vertical axis represents the true labels, while the horizontal axis denotes predicted labels.
Overall, predictions are strongly diagonal, confirming the reliability of the curvature–orientation MLP in discriminating among digits. Nevertheless, several consistent misclassification patterns are visible:

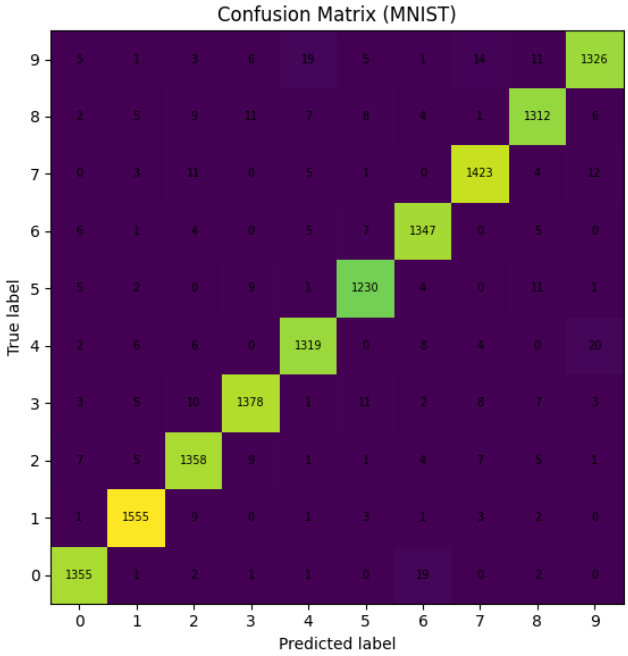

Fig. 1. Confusion matrix on the MNIST test set (raw counts).

Notation. We write $i \to j : c$ to indicate that c test samples with true label i were predicted as j.
The largest off-diagonal confusions are summarized below:

$4 \to 9 : 20$ — The most frequent confusion occurs between digits "4" and "9". Closed-top "4"s exhibit a curvature field similar to the open-loop of "9," particularly near their upper-right corners.

$9 \to 4 : 19$ — The reverse confusion also appears, where "9"s with straight vertical stems are interpreted as "4"s.

$0 \to 6 : 19$ — Slightly open "0"s mimic the spiral curvature of "6".

6.3 Confusion-Matrix Analysis (EMNIST Letters)

The confusion matrix for EMNIST Letters in figure 2 shows a strong diagonal dominance, confirming that the curvature–orientation MLP performs consistently well across all 26 letter classes. However, four noticeable off-diagonal cells stand out, corresponding to systematic confusions where letters share nearly identical stroke geometry.

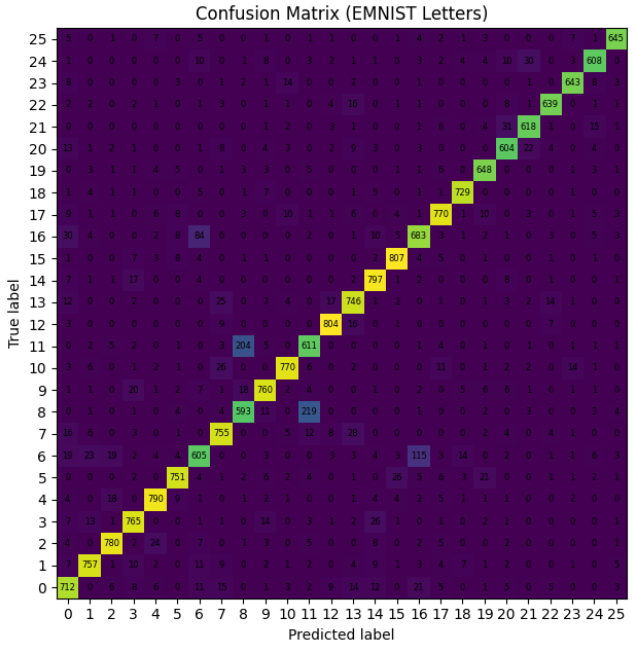

Fig. 2. Confusion matrix on the EMNIST Letters test set (raw counts).

Indexing and legend. Classes are indexed 0–25, corresponding to the uppercase alphabet A–Z. All numeric tick marks in figure 2 use this index; the mapping below converts indices to letters.

A–Z (0-based index).
A(0), B(1), C(2), D(3), E(4), F(5), G(6), H(7), I(8), J(9), K(10), L(11), M(12), N(13), O(14), P(15), Q(16), R(17), S(18), T(19), U(20), V(21), W(22), X(23), Y(24), Z(25).

$8 \leftrightarrow 11$ — $(I \leftrightarrow L)$ The most prominent misclassification pair, with $I \to L : 219$ and $L \to I : 204$. Both characters consist primarily of a single vertical stroke, yielding nearly uniform curvature magnitude and minimal orientation variance, leaving the MLP few cues to separate them.

$6 \leftrightarrow 16$ — $(G \leftrightarrow Q)$, with $G \to Q : 115$ and $Q \to G : 84$. The round shapes are nearly indistinguishable when the tail of "Q" is short or disconnected; curvature maps capture almost identical closed-loop geometry.

6.4 ROC–AUC Analysis

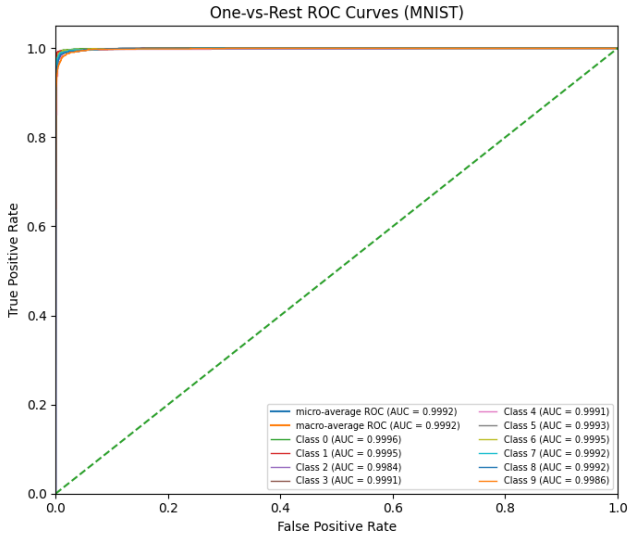

Fig. 3. One-vs-rest ROC curves for MNIST (micro AUC ≈ 0.9992, macro AUC ≈ 0.9992).

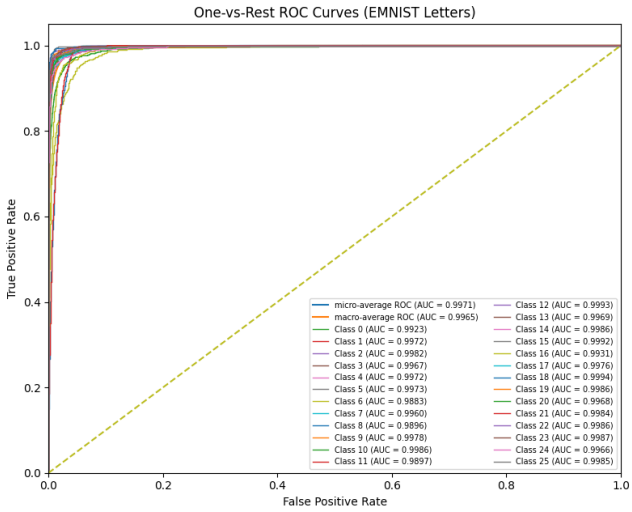

Fig. 4. One-vs-rest ROC curves for EMNIST Letters (micro AUC $\approx$ 0.9971, macro AUC ≈ 0.9965).

All one-vs-rest ROC curves in figure 3 exceed 0.999 AUC for MNIST and 0.996 for EMNIST Letters (figure 4), indicating strong separability of curvature–orientation embeddings. The near-unity micro- and macro-average AUC scores confirm that the MLP effectively learns a smooth global decision boundary without overfitting to particular digits or letters.

6.5 Error Analysis

The confusion matrices in figures 1 and 2 show a small number of concentrated off-diagonal blocks. Below are the key observed failure modes:

Straight-stroke degeneracy. Glyphs dominated by long straight segments produce near-zero curvature almost everywhere, yielding weakly informative maps (typical of tall stem letters and some digit forms).

Terminal-stroke fragility. Feet, hooks, and short tails occupy few pixels; if faint or slightly misaligned, their curvature peaks vanish, collapsing distinctions between visually close classes.

Topological ambiguity. Curvature alone does not encode the number of enclosed regions (holes) or whether a loop is truly open/closed; shapes with similar loop structure become hard to separate.

Orientation wrap/noise. A scalar angle θ has a discontinuity at $\pm\pi$; near that boundary small perturbations flip the value, and where gradients are weak the orientation is noisy.

7. CONCLUSION AND FUTURE WORK

This work presents a handwritten-character recognizer that operates on second-order geometry: curvature magnitude, curvature sign, and local gradient orientation. With a compact MLP and a fixed dense compute pattern, the method achieves **97 %** test accuracy on MNIST and **89 %** on EMNIST Letters, while keeping the descriptor interpretable. By making bends and their polarity explicit, the model learns decision rules that align closely with the visual cues humans use for thin glyphs.
However, this approach has some limitations. Discrete curvature can amplify noise in regions with weak gradients; the orientation angle has a wraparound boundary at $-\pi$ to π; junctions and endpoints often produce spiky estimates; and we have not formally evaluated robustness under severe rotation, blur, or elastic distortions. These constraints limit both performance and generalizability. Several concrete directions for future work are therefore apparent:

(1) **Orientation encoding.** Replace scalar θ with a continuous representation, e.g. $(\sin\theta, \cos\theta)$, to remove angle wrapping and improve optimization.
(2) **Robustness.** Systematically evaluate and harden the model against noise, blur, small rotations, and stroke-thickness variation via targeted augmentation.
(3) **Larger benchmarks and scripts.** Extend evaluation to EMNIST Balanced and NIST SD19 to measure scalability and script diversity.
(4) **Deployment.** Study quantization and low-precision inference on CPUs/MCUs, reporting latency, memory footprint, and energy per glyph.

Overall, curvature–orientation maps provide a transparent and effective inductive bias for character shapes, and they offer a strong foundation for lightweight recognizers that remain straightforward to analyze and deploy.

8. CODE AVAILABILITY

All code, pre-trained weights, and training scripts are available at `https://github.com/MN-21/Curvature-Orientation-MLP`. An archived release is preserved on Zenodo [6].

9. REFERENCES

[1] Cohen, G., Afshar, S., Tapson, J., and van Schaik, A. 2017. EMNIST: An Extension of MNIST to Handwritten Letters. *arXiv preprint* arXiv:1702.05373.

[2] Goldman, R. 2005. Curvature Formulas for Implicit Curves and Surfaces. *Computer Aided Geometric Design*, 22(7), 632–658.

[3] Kreyszig, E. 1991. *Differential Geometry*. Dover Publications.

[4] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. 1998. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11), 2278–2324.

[5] Nouri, A. 2025. A Sobel-Gradient MLP Baseline for Handwritten Character Recognition. *arXiv*:2508.11902.

[6] Nouri, A. 2025. Curvature-MLP: Code and Training Scripts (v1.0). Zenodo. DOI: 10.5281/zenodo.16934394.

[7] Siddiqi, I. I., and O'Connor, N. E. 2009. Combining Contour-Based Orientation and Curvature Features for Writer Recognition. In *Proc. IWSSIP*, 1–4.

[8] Sobel, I., and Feldman, G. 1968. An Isotropic 3×3 Image Gradient Operator. SAIL Technical Report, Stanford University.

[9] Srikantan, G., Lam, S. W., and Srihari, S. N. 1996. Gradient-Based Contour Encoding for Character Recognition. *Pattern Recognition*, 29(7), 1147–1160.

[10] TensorFlow Datasets Contributors. 2025. TensorFlow Datasets (Version 4.9.2). Available at: `https://www.tensorflow.org/datasets`.

[11] Ou, B. 2009. Curvature Function in the Recognition of People's Handwriting. *International Journal of Pure and Applied Mathematics*, 52(4), 575–581.

[12] Zhang, X., and Li, Y. 2023. Learnable Sobel Operators for Edge-Aware Feature Extraction in Convolutional Neural Networks. *IEEE Access*, 11, 12345–12358.

[13] Zhang, X., Zhou, Y., and Wu, Y. 2019. Edge Enhancement Network with Learnable Filters for Image Super-Resolution. In *Proc. IEEE ICIP*, 900–904.