Handwriting Trajectory Recovery of Latin Characters with Deep Learning: A Novel Exploring the Amount of Points per Character and New Evaluation Method

Simone Bello Kaminski Aires[a], Erikson Freitas de Morais[b] and Yu Han Lin[c]
Department of Computing, Federal Technology University of Parana, Washington Subtil Chueire, Ponta Grossa, Brazil

Keywords: Deep Learning, Handwriting Trajectory Recovery, Handwriting Reconstruction.

Abstract: The research on handwriting trajectory recovery (HTR) has gained prominence in offline handwriting recognition by utilizing online recognition resources to simulate writing patterns. Traditional approaches commonly employ graph-based methods that skeletonize characters to trace their paths, while recent studies have focused on deep learning techniques due to their superior generalization capabilities. However, despite promising results, the absence of standardized evaluation metrics limits meaningful comparisons across studies. This work presents a novel approach to recovering handwriting trajectories of Latin characters using deep learning networks, coupled with a standardized evaluation framework. The proposed evaluation model quantitatively and qualitatively assesses the recovery of stroke sequences and character geometry, providing a consistent basis for comparison. Experimental results demonstrate the significant influence of the number of coordinate points per character on deep learning performance, offering valuable insights into optimizing both evaluation and recovery rates. This study provides a practical solution for enhancing HTR accuracy and establishing a standardized evaluation methodology.

1 INTRODUCTION

Handwriting recognition involves transforming handwritten graphical marks into digital signals, enabling the interpretation of unique individual writing styles (Plamondon and Srihari, 2000). However, the variability in handwriting presents significant challenges, particularly in generalizing across different scripts and languages. Neural networks have shown considerable potential in addressing these challenges by learning to adapt to diverse styles and input formats (Xiong, Dai, and Meng, 2023; Gautam and Singh, 2022; Shaji, Shoba, Jemimma, and Chester, 2023). Nevertheless, offline handwriting recognition remains less accurate than online recognition due to the lack of temporal and dynamic information (Zhang, Bengio, and Liu, 2017).

To overcome this limitation, handwriting trajectory recovery (HTR) systems have been developed to reconstruct the temporal order of strokes from static images, providing dynamic information that helps distinguish characters with similar geometric forms (Noubigh and Kherallah, 2016). Traditional HTR methods often rely on heuristic approaches, such as graph-based skeletonization and greedy algorithms, to approximate handwriting trajectories (Dinh, Yang, Lee, Kim and Do, 2016). Although these methods can be effective for simpler cases, they lack the flexibility required to manage the complexity and variability inherent in handwriting.

Deep learning techniques, including Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks, have significantly advanced the field of HTR by leveraging their generalization capabilities to predict stroke sequences from static handwriting images (Singh, Rohilla and Sharma, 2024; Jayanna, Nagaraja, Yadava, Deekshith, Seelam and Jamkhandi, 2024; Lv, 2023; Qu, 2024). Recent studies have demonstrated the successful integration of CNNs with LSTMs to predict coordinate points, achieving

[a] https://orcid.org/ 0000-0003-3346-2693
[b] https://orcid.org/ 0000-0002-0845-7457
[c] https://orcid.org/ 0000-0002-4905-5462

Aires, S. B. K., Freitas de Morais, E. and Lin, Y. H.
Handwriting Trajectory Recovery of Latin Characters with Deep Learning: A Novel Exploring the Amount of Points per Character and New Evaluation Method.
DOI: 10.5220/0013383100003912

In *Proceedings of the 20th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2025) - Volume 3: VISAPP*, pages 855-862
ISBN: 978-989-758-728-3; ISSN: 2184-4321

promising results (Zhao, Yang, and Tao, 2019; Zhang, Bengio and Liu, 2017; Wang, Sonogashira, Hashimoto and Iiyama, 2019; Elbaati, Hamdi and Alimi, 2019). However, the absence of standardized evaluation methods is a limitation, as current research relies on diverse proprietary metrics or statistical coefficients with non-uniform thresholds (Kumar, Bhowmick, Bhunia, Konwer, Banerjee, Roy and Pal, 2018; Zhao, Yang and Tao, 2019; Elbaati, Hamdi and Alimi, 2019). This inconsistency hinders the comparability of results across different HTR systems.

In this context, the present study introduces a novel HTR approach leveraging deep learning, specifically targeting Latin lowercase characters from the IRONOFF dataset. Unlike existing methods, this work proposes a standardized evaluation framework to assess the accuracy of predicted trajectories, enabling consistent and comparable analyses. Furthermore, the study systematically examines the impact of varying the number of coordinate points per character on model performance, offering valuable insights for optimizing trajectory recovery and evaluation methodologies.

The remainder of this paper is organized as follows: Section 2 reviews related work on HTR, Section 3 details the methodology and experimental setup, Section 4 discusses the proposed evaluation framework, Section 5 presents results and analysis, and Section 6 concludes and future directions.

2 RELATED WORKS

Handwriting trajectory recovery (HTR) methods can be broadly categorized into two main approaches: heuristic algorithms and artificial intelligence/machine learning-based techniques. Heuristic algorithms, such as skeletonization and graph-based pathfinding, have been widely utilized to approximate handwriting trajectories (Noubigh and Kherallah, 2016; Nguyen and Blumenstein, 2010). Prominent examples include the use of Euclidean path optimization in undirected graphs (Nagoya and Fujioka, 2011) and code chain algorithms to recover trajectories from character skeletons (Sharma, 2013). Subsequently, (Sharma, 2015) extended this approach by integrating dynamic and static features, significantly improving handwriting recognition performance.

Genetic algorithms have also been employed to optimize trajectory paths, as demonstrated by (Elbaati, Kherallah, Ennaji and Alimi, 2009), who restored stroke chronology for Arabic handwriting recognition. More recently (Jin, Ran, Yuan, Lv, Wang and Xiao, 2024) introduced the Bagging in Hidden Semi-Markov Model (BHSMM) algorithm, which partitions demonstration data into sub-datasets, encodes them using Hidden Semi-Markov Models (HSMM), and derives task-specific weights to enhance trajectory accuracy and robustness.

Deep learning has emerged as a transformative approach in HTR, leveraging convolutional neural networks (CNNs) and recurrent neural networks (RNNs) for feature extraction and sequence prediction. In work present by (Zhao, Yang and Tao, 2019) are developed a dual-CNN architecture to extract static and dynamic writing energies, combining them to recover the drawing order. Similarly (Wang, Sonogashira, Hashimoto and Iiyama, 2019) utilized a hybrid strategy with CNNs and graph-cut algorithms to refine stroke order in Chinese characters. Both approaches highlight the integration of deep learning with heuristic techniques to enhance performance.

Fully deep learning-based methods have also been explored (Kumar, Bhowmick, Bhunia, Konwer, Banerjee, Roy and Pal, 2018) proposed a CNN-LSTM framework to generalize dynamic feature learning across scripts, while (Elbaati, Hamdi and Alimi, 2019) utilized a VGG-16 combined with a beta-LSTM for trajectory recovery. However, these studies lack consistent evaluation metrics, relying on statistical measures such as Dynamic Time Warping (DTW) and Root Mean Square Error (RMSE), which require arbitrary thresholds. Others, like (Kumar, Bhowmick, Bhunia, Konwer, Banerjee, Roy and Pal, 2018), adopted parameters from (Rousseau, Anquetil and Camillerapp, 2005), or bypassed these metrics entirely by integrating predicted trajectories into online recognizers (Elbaati, Hamdi and Alimi, 2019).

While these methods represent significant advancements, the lack of a standardized evaluation framework hinders comparability across studies. To address this limitation, our approach proposes a universal evaluation system capable of assessing both the geometric accuracy of predicted characters and the correctness of coordinate sequences. This framework ensures applicability and enhances the reliability and consistency of HTR evaluations.

3 METODOLOGY

This section outlines the process of converting offline data into online trajectories, detailing the dataset, preprocessing, neural network architecture, training process, and evaluation methods. The proposed deep

neural network was inspired by the architecture proposed by (Kumar, Bhowmick, Bhunia, Konwer, Banerjee, Roy and Pal, 2018). We use their HTR system as a start point experiment to perform the handwriting recovery and develop our own deep learning network to recover the handwriting characters trajectory to make a benchmark.

3.1 Dataset

For the handwriting trajectory recovery task in this work, the dataset must have both on/off information. The experiments of this work are carried on the online data of Latin IRONOFF (Viard-Gaudin, Lallican, Knerr and Binter, 2015), the images are the drawings of online coordinates.

The IRONOFF database contains both on-line and offline information for each character. It's composed by 4.086 isolated digits and 21.364 isolates characters. Our experiments only use the single strokes of those isolates characters, which constitutes a quantity of 10.685 characters. Seven datasets with different levels of granularity were created, varying the number of points per character to 20, 30, 40, 50, 60, 70, and 100 points.

3.2 Pre-Process

As mentioned before, we only use the on-line data from the database, more specifically, the (x, y) coordinates. The pre-processing stage are done in those coordinates, and is following by three main step: resizing, normalization and data augmentation.

3.2.1 Resizing

The goal is to resize the characters to 64x64. First, the region of interest is translated by shifting the main character to the origin (0, 0). This is achieved by subtracting the smallest coordinate value from all coordinate values.

Afterward, the resizing is computed as follows. Assuming C represents all the coordinates of a character and C(xi,yi) is a single coordinate from C, the scaling ratios for the x and y axes are rx=W/64r and ry=H/64, where W and H are the highest coordinate values along the x and y axes. The new coordinates are then calculated using **Equation 1**.

$$S(x_i, y_i) = (\frac{x_i}{rx}, \frac{y_i}{ry}) \quad (1)$$

With that, the coordinates will be within the limit of 64x64.

3.2.2 Normalization

In this phase, we want to standardize the amount of the coordinates (x, y) of all samples. By applying the Bresenham Algorithm (Bresenham, 1965), we generate new points by connecting two coordinates.

We do this to all the coordinate pairs, this will create the skeleton of the character. Then, the points are sampled uniformly over the complete skeleton trajectory by 20, 30, 40, 50, 60 and 70 points (points per character). This will produce seven different databases, each one will feed their own deep neural network, that are used in the experiments.

3.2.3 Data Augmentation

Deep learning neural networks require large datasets to achieve effective generalization (LeCun, Bengio, and Hinton, 2015). Data augmentation enhances dataset size and quality by applying transformations to images, improving the network's performance (Shorten and Khoshgoftaar, 2019).

For handwriting trajectory recovery (HTR), careful selection of augmentation algorithms is crucial to avoid harming the learning process. Techniques like random erasing are unsuitable due to limited data points, while transformations such as color adjustments or noise injection are irrelevant due to pre-processed, binarized images. Slight rotations, however, are effective for character recognition tasks, as they introduce variability in the dataset by rotating samples between 1° and 359° (Shorten and Khoshgoftaar, 2019). For HTR, this involves rotating data points within each sample to enhance learning.

In Figure 1, the coordinates are drawn for visualize the effect of rotation. We apply the rotation on an central axis in four different angles: 30◦, 15◦, −15◦ and −30◦ (1). With the original sample at 0◦, we expand the dataset in five times compared the original data amount.

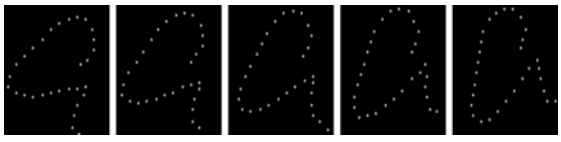

Figure 1: Illustration of Rotation Augmentation.

3.3 Deep Neural Network

The two main networks used in this work are the combination of CNN and Encoder- Decoder LSTM, with a fully connected network with one layer and two output. The offline images is the input of the CNN, which have has main task extract the high level

features of the characters. The set of features extracted from the CNN are the high-level representation of the trajectory of the character, that are the input of the encoder-decoder LSTM network. The encoder-decoder LSTM network will interpret and translate the features in a dependency string, which are the input of a fully connected layer, that will produce the (x, y) coordinates (Figure 2).

In Figure 2, are presented the coordinates prediction are done by a set of iterations. For each iteration, a new coordinate is predicted and queued behind the coordinates that are already has been predicted. At the beginning, the set of coordinates are empty. The second iteration already have the coordinate of the first iteration, so the Decoder LSTM will know that he has to predict the next coordinate following.

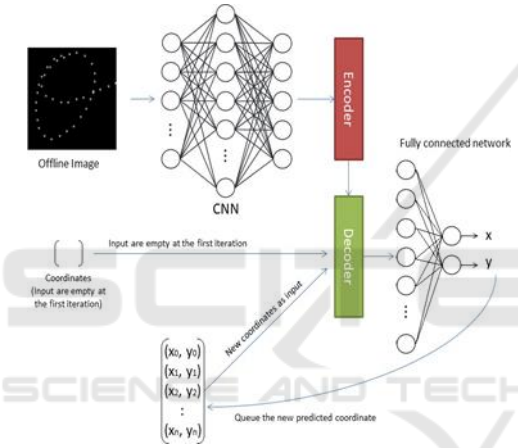

Figure 2: CNN and Encoder-Decoder LSTM.

The numbers of iterations are the number of points of the dataset. The main reason to normalize the dataset to a fixed number of points is to increase the learning process. If the network has to learn different number of points from each sample, different weights of the Decoder LSTM network will needed to be refreshed; if the amount of points are the same, the same weights for all samples are used.

We use the L1 distance to compute the fitness of the network as follows:

$$L = \frac{1}{n}\sum_{t=1}^{n} ||\vec{z_t} - \vec{Z_t}|| \quad (2)$$

In Equation 2, where $\vec{z_t}$ is the predicted coordinate, $\vec{Z_t}$ is the ground-truth coordinate and n the amount of points of each sample.

3.3.1 Networks Parameters

We explore some variations of (Kumar, Bhowmick, Bhunia, Konwer, Banerjee, Roy and Pal, 2018) network by adding layers in both CNN and Encoder-Decoder LSTM. We reach results that better fit for the HTR problem on latin characters.

The network proposed by (Kumar, Bhowmick, Bhunia, Konwer, Banerjee, Roy and Pal, 2018) have six convolutional layers (CL) and two bidirectional layers in each Encoder-Decoder LSTM, we call netBh. Four additional configurations are tested in the experiments. We call them net-v1, net-v2, net-v3 and net-v4, respectively.

- **netBh:** 6 CL and 2 LSTM layers.
- **net-v0:** 6 CL and 3 LSTM layers.
- **net-v1:** 8 CL and 3 LSTM layers.
- **net-v2:** 12 CL and 3 LSTM layers.
- **net-v3:** 16 CL and 3 LSTM layers.

These architectures were tailored to determine the effectiveness of deeper networks in processing datasets with higher point densities per character.

3.3.2 Implementation Details

All the experiments are conducted on a serve with Nvidia Ge-Force GTX Titan 6GB, i7-3770K CPU and 8GB of memory. All coding with Python and the Tensorflow framework.

The networks are trained with 200 epoch with cross-validation. Samples are divided in 70% training, 15% validation and 15% for test. The time of training is directly proportional to the amount of points of the dataset.

3.4 Evaluation Method

Evaluation methods in the literature often use statistical metrics such as DTW, RMSE, and Pearson Correlation to measure the similarity between predicted coordinates and ground truth. While these metrics indicate geometric similarity, they rely on arbitrary thresholds that affect final accuracy, leading to inconsistent results. Some metrics also include stroke direction evaluation, but they lack standardization.

The evaluation method described by (Kumar, Bhowmick, Bhunia, Konwer, Banerjee, Roy and Pal, 2018) is less inconstant than those mentioned before. By formalizing the first evaluation proposed by (Rousseau, Anquetil and Camillerapp, 2005), the metric can compute the accuracy by:

- Starting Point (SP): Evaluates whether the starting point of the sequence is correctly predicted.
- Junction Points (JP): Measures accuracy in determining the entry and exit paths at junction points.
- Complete Trajectory (CT): Verifies if the entire trajectory, including the starting point, is correctly predicted.

Before applying those metrics, (Kumar, Bhowmick, Bhunia, Konwer, Banerjee, Roy and Pal, 2018) also translate the output coordinate points of the network to the nearest point on the ground truth skeleton as a post processing step. In the research for a standard evaluation measure, we could deduce that the last one has a better potential to reflect the better accuracy for the predicted results. However, in a qualitative analysis, the experiments show the inconsistency of the metric by assigns false positives and false negatives to the predicted coordinate sequence. The reason of these factors will be described below.

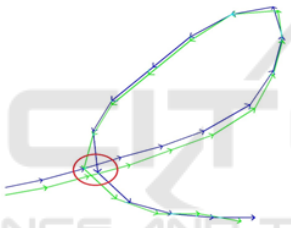

Figure 3: Trajectory of the ’e’ character.

In Figure 3, the predicted trajectory (blue) has been a little bit deviate from the ground-truth (green). First of all, the post processing step is inaccurate. We can see the case illustrated at Figure 3. The translation step translates the crossing coordinates (indicated in red) wrongly. We can see that the third coordinate are not translated to the correct place, and it will be considered a wrong prediction by the evaluation method. But the prediction are not wrong, it just a little bit deviated. This is a case of false negative and occurs with various predicted samples.

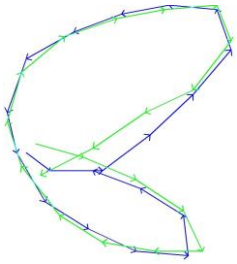

Figure 4: Trajectory of the ’p’ character.

In the learning process, the network received a specific coordinate order to learn. The test sample are the same character, but the coordinate points are ordered in the opposite order. The network can predict what he learned and can estimate the geometric form of the character. The coordinate points sequence are inverse, but it is still a right prediction. The example is show at Figure 4.

Translating the predicted points to the nearest ground-truth coordinates can severely distort the geometric shape of the predicted trajectory, leading to an incorrect relationship between the prediction and ground truth, as shown in Figure 5. This results in a false positive, where the evaluation method classifies the prediction as correct due to the correct stroke direction, even though the geometric shape of the character is inaccurate.

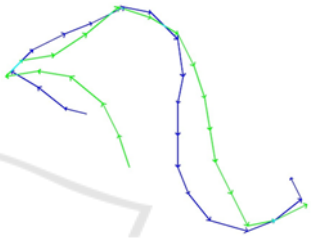

Figure 5: Trajectory of the ’r’ character.

By analysis the samples, the need of a precise evaluation method to evaluate HTR deep learning systems was noted. The last method mentioned compute 15% samples of the test dataset incorrectly (false positives and negatives), so the final accuracy of the network are not accurate enough. The next section, we will present a new evaluation method that overcome the current scenario of the reliability lack from HTR evaluation accuracy.

3.5 Proposed Evaluation Method

In this section, we present our proposed evaluation method to evaluate the HTR deep learning systems.

To compute the evaluation of a HTR system, the predicted online points have to be compared with the online ground truth. Since the deep learning networks prediction are slightly deviated from the ground-truth, the comparison with them become a not simple task. Our method can overcome this situation by using a kernel to combine the predicted coordinate with the ground-truth skeleton and output the correct result.

The evaluation process is described below.

The ground-truth skeleton is divided into N segments, where the choice of N significantly impacts the character's geometric representation. A high N

value minimizes the impact of removing a single segment, preserving the character's shape, while a low N value causes greater deformation, potentially rendering the character unrecognizable. Therefore, N directly affects the final accuracy. Experiments reveal that the higher the points per character (p/c), the higher the N value required. A ratio between p/c and N is established using a defined computation method, as outlined in the Equation 3.

$$N = \left\lfloor \sqrt{p/c} * 2 \right\rfloor \quad (3)$$

In Equation 3, N is the number of segments a character will be divided into, and p/c denotes the number of points per character. The floor function is applied to convert the result into the largest integer less than or equal to the computed value.

Once segmentation is complete, the character's geometric form is evaluated by checking if the predicted coordinates align with the ground-truth coordinates. This is done using a kernel-based technique, where the kernel slides along the ground-truth skeleton to verify the presence of predicted points in each segment. While Zhao, Yang, and Tao (2019) use kernels referred to as top-1, top-5, and top-10, this work employs a 5x5 kernel, equivalent to top-25, to accommodate the larger image sizes used here. The process, illustrated in Figure 6, ensures a comprehensive evaluation of predicted trajectories.

In Figure 6, the white dots represent the predicted coordinate points outputted by the deep neural network. Each segment is colored by better visualization. If the kernel found a predicted point, the segment that the kernel is following is considered predicted. If one of the segments of the character are not predicted (in other words, no predicted points were found in the sliding kernel covered area), means that the network failed to predict the character.

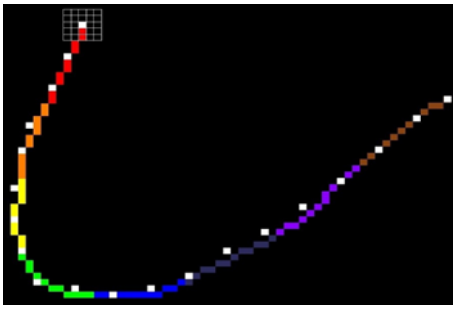

Figure 6: 5x5 Kernel sliding the skeleton ground-truth.

The evaluation process ensures that predicted points cover all segments of the original skeleton. If a predicted point overlaps with two segments, it is assigned to the segment that lacks a predicted point. Once all segments are covered, the predicted points are evaluated for their order.

As shown in Figure 7, segments are evaluated for correct ordering by checking if at least one predicted point from each segment aligns with the segment's sequence order. If such a combination exists, the prediction order is considered correct. Multiple combinations of predicted points and segments may be valid.

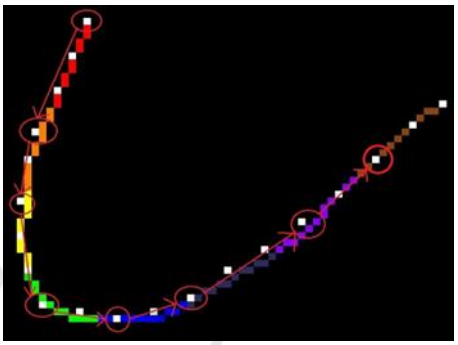

Figure 7: Segments ordering.

In Figure 7, white dots represent predicted coordinate points, and arrows illustrate an example of correct ordering identified by the method. A prediction is considered correct if it covers the geometric shape of all segments from the ground-truth skeleton and preserves the correct order of these segments. This approach resolves the false positives and false negatives identified in earlier methods.

4 RESULTS

We evaluate our system by using the existing evaluation method (Complete Trajectory - CT) and the proposed evaluation method. The metrics are applied in all the data sets generated by the pre-processing step. Table 1 and 2 show the accuracy of the outputs from the network configuration of (Kumar, Bhowmick, Bhunia, Konwer, Banerjee, Roy and Pal, 2018) in all seven data sets.

Table 1: Accuracy from evaluation methods – part 1.

	20*p/c*	30*p/c*	40*p/c*
Evaluation CT	84,6%	79,2%	71,9%
Proposed	91,4%	**91,6%**	**87,8%**

Table 2: Accuracy from evaluation methods – part 2.

	50p/c	60p/c	70p/c	100p/c
Evaluation CT	67,8%	56,1%	49,3%	32,0%
Proposed	**86,0%**	**83,7%**	**79,5%**	**39,7%**

The proposed evaluation can correct the false positives and false negatives generated by the CT method. The results show that the predictions are quite better with our evaluation method. Six of seven data sets show a very significant amount of improvement. In the CT evaluation, the 20p/c and 70p/c have a difference of 36 points percentage, while our evaluation method have only 12 points. Such difference means that the CT evaluation doesn't compute with efficiency the predicted coordinates with more points per character.

The Table 3 and Table 4 show all the network developed in this work in all data sets in the two evaluation methods.

Table 3: Accuracy from evaluation methods off all networks – part 1.

Dat Set	net-v1	net-2
20*p/c*	85,29% - **90,75%**	85,17% - **95,31%**
30*p/c*	79,20% - **94,34%**	81,18% - **95,14%**
40*p/c*	74,33% - **92,51%**	72,27% - **91,06%**
50*p/c*	68,02% - **88,56%**	66,42% - **87,97%**
60*p/c*	60,84% - **86,59%**	54,84% - **85,95%**
70*p/c*	46,02% - **83,02%**	55,04% - **86,93%**
100*p/c*	29,87% - **44,82%**	39,55% - **45,13%**

Table 4: Accuracy from evaluation methods off all networks – part 2.

Dat Set	net-v3	net-v4
20*p/c*	85,01% - **90,79%**	81,89% - **85,17%**
30*p/c*	79,23% - **92,39%**	76,00% - **76,97%**
40*p/c*	73,05% - **89,86%**	67,81% - **76,42%**
50*p/c*	64,11% - **86,46%**	57,92% - **65,75%**
60*p/c*	56,19% - **81,01%**	51,57% - **59,24%**
70*p/c*	48,37% - **78,27%**	44,68% - **50,36%**
100*p/c*	25,70% - **45,45%**	25,28% - **23,29%**

Is noticeable the accuracy difference between the evaluations. We have improve the accuracy considerably for the 70p/c data set in the net-v1, net-v2 and net-v3 networks. Besides that, we show that by resizing the amount of points per character and by providing the network the right number of layers of the network, the HTR can achieve better results, as shown the result by 20p/c on net-v2 network.

5 CONCLUSION

The paper introduces a new evaluation method for HTR systems that are more accurate comparing the existing method. The proposed method utilizes the kernel sliding system that can check if the predicted points estimate the geometric form of the original character, and the segmentation system simplifies the evaluation of trajectory order sequence. The experiments on the IRONOFF database also shown that by normalizing and resizing the amount of points per character can facilitate the network learning. Our better results lay on the net-v2 on 20p/c, which achieve 95,31% of accuracy.

Our great achieve it's we provide a huge improvement in the accuracy by proposing a more efficient evaluation system, and a system to improve the learning rate of the network, which is a important and very challenge topic. Our network and evaluation method effectively recover the trajectory of handwriting from offline images of Latim characters.

In conclusion, Handwriting Trajectory Recovery (HTR) is a key advancement in handwriting recognition, transforming static images into dynamic temporal data. Its versatility spans handwriting recognition, robotic writing, historical manuscript preservation, accessibility, and forensic analysis. By recovering stroke sequences and geometric structures, HTR enhances accuracy and enables applications in multilingual systems and digital design. Continued improvements in HTR methods and evaluation will further expand its impact, addressing complex handwriting challenges with greater precision and adaptability.

ACKNOWLEDGEMENTS

The research leading to these results has received support from the CAPES, Araucaria Foundation and UTFPR-PG.

REFERENCES

Bresenham, J. E. (1965). Algorithm for computer control of a digital plotter. IBM Systems Journal, 4(1), 25–30. https://doi.org/10.1147/sj.41.0025

C. Shaji, V. Betcy Thanga Shoba, T. Jemimma, J. Robert Edwin Chester (2023). Handwriting Recognition Using Artificial Intelligence with Neural Network. ICRICC 23. https://doi.org/10.59544/tjer7283/icricc23p4

Dinh, M., Yang, H.-J., Lee, G.-S., Kim, S.-H., & Do, L.-N. (2016). Recovery of drawing order from multi-stroke

English handwritten images based on graph models and ambiguous zone analysis. Expert Systems with Applications, 64, 352–364. https://doi.org/10.1016/j.eswa.2016.08.004

Elbaati, A., Hamdi, Y., & Alimi, A. M. (2019). Handwriting recognition based on temporal order restored by the end-to-end system. In 2019 International Conference on Document Analysis and Recognition (ICDAR) (pp. 1231–1236). IEEE. https://doi.org/10.1109/ICDAR.2019.00199

Elbaati, A., Kherallah, M., Ennaji, A., & Alimi, A. M. (2009). Temporal order recovery of the scanned handwriting. In 2009 10th International Conference on Document Analysis and Recognition (pp. 1116–1120). IEEE. https://doi.org/10.1109/ICDAR.2009.232

Gautam, K., & Singh, S. (2022). Neural Network to Recognize Handwriting Objects.

Jin, Y., Ran, T., Yuan, L., Lv, K., Wang, G., & Xiao, W. (2024). Bagging no modelo semi-Markov oculto para geração de trajetória de robô de escrita manual. J. Intell. Fuzzy Syst., 46, 6325-6335. https://doi.org/10.3233/jifs-237275

KumarBhunia, A., Bhowmick, A., Bhunia, A. K., Konwer, A., Banerjee, P., Roy, P. P., & Pal, U. (2018). Handwriting trajectory recovery using end-to-end deep encoder-decoder network. In 2018 24th International Conference on Pattern Recognition (ICPR) (pp. 3639–3644). IEEE. https://doi.org/10.1109/ICPR.2018.8545898

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444. https://doi.org/10.1038/nature14539

Nagoya, T., & Fujioka, H. (2011). A graph theoretic algorithm for recovering drawing order of multi-stroke handwritten images. In 2011 Third International Conference on Intelligent Networking and Collaborative Systems (pp. 569–574). IEEE. https://doi.org/10.1109/INCoS.2011.144

Nguyen, V., & Blumenstein, M. (2010). Techniques for static handwriting trajectory recovery: A survey. In Proceedings of the 9th IAPR International Workshop on Document Analysis Systems (pp. 463–470). ACM. https://doi.org/10.1145/1815330.1815382

Noubigh, Z., & Kherallah, M. (2017). A survey on handwriting recognition based on the trajectory recovery technique. In 2017 1st International Workshop on Arabic Script Analysis and Recognition (ASAR) (pp. 69–73). IEEE. https://doi.org/10.1109/ASAR.2017.8067766

Plamondon, R., & Srihari, S. N. (2000). Online and off-line handwriting recognition: A comprehensive survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(1), 63–84. https://doi.org/10.1109/34.824821

Rousseau, L., Anquetil, E., & Camillerapp, J. (2005). Recovery of a drawing order from off-line isolated letters dedicated to on-line recognition. In Eighth International Conference on Document Analysis and Recognition (ICDAR'05) (pp. 1121–1125). IEEE. https://doi.org/10.1109/ICDAR.2005.123

Sharma, A. (2013). Recovery of drawing order in handwritten digit images. In 2013 IEEE Second International Conference on Image Information Processing (ICIIP-2013) (pp. 437–441). IEEE. https://doi.org/10.1109/ICIIP.2013.6707642

Sharma, A. (2015). A combined static and dynamic feature extraction technique to recognize handwritten digits. Vietnam Journal of Computer Science, 2(3), 133–142. https://doi.org/10.1007/s40595-015-0041-3

Sharma, N., & Agarwal, P. (2018). Offline handwriting recognition using neural networks.

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. Journal of Big Data, 6(1), 60. https://doi.org/10.1186/s40537-019-0197-0

Viard-Gaudin, C., Lallican, P. M., Knerr, S., & Binter, P. (1999). The ireste on/off (ironoff) dual handwriting database. In Proceedings of the International Conference on Document Analysis and Recognition (pp. 455–458). https://doi.org/10.1109/ICDAR.1999.791781

Wang, Y., Sonogashira, M., Hashimoto, A., & Iiyama, M. (2019). Two-stage fully convolutional networks for stroke recovery of handwritten Chinese character. In Asian Conference on Pattern Recognition (pp. 321–334). Springer. https://doi.org/10.1007/978-3-030-04793-4_26

Xiong, Y., Dai, Y., & Meng, D. (2023). Deep Frame-Point Sequence Consistent Network for Handwriting Trajectory Recovery. 2023 IEEE 29th International Conference on Parallel and Distributed Systems (ICPADS), 2151-2158. https://doi.org/10.1109/ICPADS60453.2023.00291

Zhang, X.-Y., Bengio, Y., & Liu, C.-L. (2017). Online and offline handwritten Chinese character recognition: A comprehensive study and new benchmark. Pattern Recognition, 61, 348–360. https://doi.org/10.1016/j.patcog.2016.07.004

Zhao, B., Yang, M., & Tao, J. (2019). Drawing order recovery for handwriting Chinese characters. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 3227–3231). IEEE. https://doi.org/10.1109/ICASSP.2019.8682696