# Supporting Chinese Character Educational Interfaces with Richer Assessment Feedback through Sketch Recognition

Tianshu Chu* Paul Taele† Tracy Hammond‡

Sketch Recognition Lab, Texas A&M University

**ABSTRACT**

Students of Chinese as a Second Language (CSL) with primarily English fluency often struggle with the language's complex character set. Conventional classroom pedagogy and relevant educational applications have focused on providing valuable assessment feedback to address their challenges, but rely on direct instructor observation and provide constrained assessment, respectively. We propose improved sketch recognition techniques to better support Chinese character educational interfaces' real-time assessment of novice CSL students' character writing. Based on successful assessment feedback approaches from existing educational resources, we developed techniques for supporting richer automated assessment, so that students may be better informed of their writing performance outside the classroom. From our evaluations, our techniques achieved recognition rates of 91% and 85% on expert and novice Chinese character handwriting data, respectively, greater than 90% recognition rate on written technique mistakes, and 80.4% f-measure on distinguishing between expert and novice handwriting samples, without sacrificing students' natural writing input of Chinese characters.

**Keywords:** Sketch recognition, Chinese, intelligent tutoring system, language learning, handwriting recognition, writing assessment

**Index Terms:** Applied computing—Education—Computer-assisted instruction; Applied computing—Document management and text processing—Online handwriting recognition

## 1 INTRODUCTION

The Chinese language continues to be a popular foreign language of study with native English users for the past several decades [29]. Educators emphasize to students that learning the language's written component is crucial to achieve greater fluency [32]. However, learning its Chinese characters is challenging to students due to a vast and complicated writing script [5]. Unlike individual English alphabet letters formed from a few lines or curves, individual Chinese characters commonly consist of more complex or diverse strokes [27].

Conventional classroom pedagogy focus on reducing students' challenges in learning Chinese characters through instructors' specialized expert feedback, such as teaching written technique strategies for easing students' learning and memorization of character writing [32]. While instructors provide valuable assessment of students' character writing performance, their teaching methods can be prone to bias [9]. Instructors can offer assessment feedback on written technique by directly observing students writing characters. On the other hand, they are generally limited to only the students' final visual structures of the written characters from classwork assigned outside the classroom or within larger classroom sizes that limit individual attention.

One area of computer-assisted language learning resources with strong potential to offer valuable detailed assessment of students' character writing performance are stylus-driven intelligent tutoring systems (ITS). Such systems that incorporate sketch recognition—or the automated recognition of hand-drawn diagrams by a computers [6]—have proven successful in other educational domains that utilize writing and sketching. Gesture and vision information from students' handwritten input can enable educational interfaces to automatically recognize and assess students' character writing intentions such as proper visual structure and written technique. This can not only reduce the time-consuming burden of instructors in directly observing students' writing performance, but can also grant students further opportunities to receive feedback and assessment in improving their character writing.

Prior sketch-based ITS interfaces that have focused on East Asian language handwriting (e.g., [26–28]) have similarly employed sketch recognition techniques to assess students' character writing performance in real-time. However, these prior interfaces remain constrained in providing more sophisticated feedback and assessment of students' Chinese handwriting. From our conducted pilot study of these prior interfaces, we observed several weaknesses that were consistent with sketch techniques employed by these interfaces: 1) lack of more robust assessment capabilities for poorly-written characters, 2) constrained feedback to only a few type of written technique mistakes, and 3) narrow focus of assessing more on proper written technique and less on proper visual structure.

In this work, we propose novel sketch recognition techniques for supporting designers and developers of Chinese character educational applications, so that their interfaces may offer valuable richer assessment and feedback of students' Chinese character writing performance. We demonstrate the capabilities of our techniques on a set of 27 characters (Table 1) that are representative of fundamental introductory characters found in novice students' Chinese language textbooks. Furthermore, we developed a preliminary writing interface that utilizes our proposed sketch recognition techniques for demonstrating the assessment performance of users' written characters. From our evaluations, our proposed techniques were more accurate and robust in recognizing the visual structures of students' characters, and provided richer real-time assessment feedback on their character's written technique. We also demonstrated that our proposed techniques can distinguish between novices' and experts' written characters, which designers and developers of Chinese character educational applications can leverage for generating helpful grading rubrics that can measure the visual quality of students' written characters.

## 2 RELATED WORK

### 2.1 Symbol Writing Intelligent Tutoring Systems

Researchers have leveraged stylus-based computing devices for developing ITS interfaces related to symbol writing. Such interfaces span a wide range of domains that consist of both language symbols, such as English (e.g., KimChi2 [12]) and non-English (e.g., Urdu

*e-mail: chutianshu@tamu.edu
†e-mail: ptaele@tamu.edu
‡e-mail: hammond@tamu.edu

Graphics Interface Conference 2018
8-11 May, Toronto, Ontario, Canada
Copyright held by authors. Permission granted to CHCCS/SCDHM to publish in print and digital form, and ACM to publish electronically.

Table 1: Representative Chinese characters assessed by our sketch recognition techniques.

| | | | | | |
|---|---|---|---|---|---|
| Eight | 八 | No | 不 | Car | 车 |
| Big | 大 | Two | 二 | Good | 好 |
| Gold | 金 | Nine | 九 | Six | 六 |
| Horse | 马 | You | 你 | Cow | 牛 |
| Female | 女 | Seven | 七 | Thousand | 千 |
| People | 人 | Three | 三 | Ten | 十 |
| Four | 四 | He | 他 | Ten thousand | 万 |
| Article | 文 | Me | 我 | Five | 五 |
| Small | 小 | One | 一 | Son | 子 |

Qaeda [19]) alphabet letters; non-language symbols, such as music (e.g., Maestoso [25]) and mathematics (e.g., MathPad2 [13]), and a mixture of symbol types [20]. More related symbol writing ITS interfaces for the domain of East Asian language scripts include the related scripts of Japanese kanji (e.g., Hashigo [27]) and Mandarin phonetic symbols (e.g., BopoNoto [26]). These latter ITS interfaces for writing East Asian language scripts—which strongly relate to Chinese characters—incorporate similar goals of providing assessment on the symbols' visual structure and written technique. However, our proposed sketch recognition techniques expand upon these interfaces' own techniques with more robust recognition and richer assessment of novice students' written East Asian language script symbols (i.e., Chinese characters).

### 2.2 Handwritten Symbol Recognition

Research areas relevant to our work that target written Chinese character recognition include those that focus on the characters' visual structure and written technique. There have been at least several decades of research work that focus on improving handwriting recognition of Chinese characters in the machine learning field such as neural networks and statistical models [24]. Early work by LeCun et al. recognized handwriting with backpropagation networks [14] that have been extended to offline recognition of handwritten Chinese characters (e.g., [16, 17]). Latter research efforts for online recognition rely on the trajectory of points, which are more relevant to our proposed recognition techniques.

Alternatively, gesture- and vision-based recognition approaches [6, 11, 30] have been explored for novice students' written East Asian language script symbols (e.g., Chinese characters) [26–28]. One of the main reasons is that these techniques have been successful in determining whether the students' written symbols are recognized as properly written, unlike conventional handwriting recognition techniques that focus on best classifying that symbol. Moreover, since Chinese characters can usually be formed from polyline shapes that lie on the eight compass directions [3], recognizers have also incorporated orientation-based features such as Gabor features (e.g., [10]) that could similarly be applied for written technique assessment..

Machine Learning and fuzzy techniques have been employed extensively on assessing the quality of handwritings [7,23], but have also relied on pixel data that reflected the overall structure of symbols with little attention to its stroke-level characteristics. Corner-finding algorithms (e.g., [8]) are one potential solution to such focus at the stroke level to extract features from writing samples, and have been explored as features for visual quality assessment of East Asian language script symbols (e.g., [26, 27]). However, these methods are limited in the types of feedback that they can provide to students for cases when their written input contains multiple types of written technique mistakes.

While much of the research work in sketch recognition have focused on diagrammatic sketches, prior research works have applied these techniques for sketches with symbol-like properties. Gesture-based sketch recognition techniques—which focus on the process of how the sketch is drawn rather than its final form [18]—have been used effectively for recognizing single-stroke (e.g., [31]) and multi-stroke gestures (e.g., [1]), respectively, and grouping gesture strokes together [21]. Geometric-based sketch recognition techniques have less constraints to the drawing process, and instead focus on the geometric features within sketches [6]. Vision-based recognition techniques solely rely on the visual structure of the sketches, which are calculated based on the location of points that are often used to measure similarities between sketches [11]. Both geometric- and vision-based methods do not require users to draw in a pre-defined manner, but instead recognize inputs from their visual stroke information. Therefore, these techniques have been applied for interfaces in educational domains where drawing techniques are less emphasized (e.g., [26, 27]).

## 3 Methodology

### 3.1 Demonstration User Interface

In Figure 1, we created a demonstration user interface to visualize how our proposed sketch recognition techniques would assist designers and developers in enhancing the assessment capabilities of their educational applications for Chinese character writing. This interface displays a question for each character from Table 1 at the top, which prompts the user to write the corresponding character accordingly. Users would then need to draw in the square writing space on the left, and correct their writing with *Undo* and *Clear* buttons to cancel their last written stroke and clear their entire writing, respectively. Upon completion of writing, the user would click the *Finish* button to prompt the interface to evaluate their input's visual structure. After the interface successfully recognizes the input symbol, it then evaluates the input's written technique performance. During this process, the user's writing mistakes will be highlighted in the original handwritten character, and instructive feedback that demonstrates proper writing will be displayed.

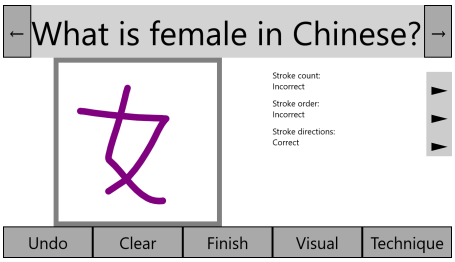



Figure 1: Demonstration interface for visualizing the assessment capabilities of our proposed sketch recognition techniques for Chinese character educational applications.

### 3.2 Handwriting Recognition

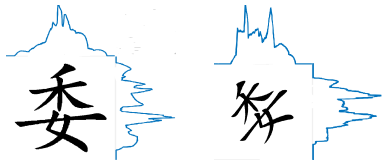
Robust handwriting recognition accuracy of our proposed sketch recognition techniques is crucial because it is a primary requirement for determining whether the user achieved proper character writing. That is, an effective recognition algorithm should accurately identify which character that the CSL learner intended to draw. This goal requires the recognition system to: 1) rely on features that

distinguish between template (i.e., model) characters, and 2) have a reasonable tolerance for novice students' common mistakes. Previous works have demonstrated that template-matching methods perform reasonably well for East Asian script symbols [26]. However, prior vision-based template-matching algorithms [11, 30] performed weakly in recognizing badly-drawn symbols. These algorithms often fail when novice users either make common mistakes in written technique, or write the correct character but with weak visual structure. To accurately recognize novice students' writing inputs with acceptable tolerance to both written technique and visual structure mistakes, we propose a novel template-matching technique that was motivated by the following assumptions:

1. Chinese characters are normally formed with multiple straight line segments, where each line segment can be approximated as being positioned at one of four orientations: horizontal, vertical, left diagonal, and right diagonal [3, 22].

2. CSL learners usually know which orientation corresponds to each straight line segment's position, and its relative sequence of each line segment on each orientation, but can place these segments with little consideration on how to balance them in the entire structure.

The first assumption implies that projections on the four orientations provide useful information about the lengths and locations of the stroke segments for each orientation.



(a) Non-diagonal projections (b) Diagonal projections

Figure 2: Projections on the four orientations for a Chinese character.

Figure 2 shows how a Chinese character with two horizontal, one vertical, and several diagonal strokes are projected at different orientations, where a projected orientation contains peaks and valleys that indicate the locations and lengths of the stroke. These features are helpful in recognizing novice users' handwriting inputs, because they typically write each character stroke by stroke, so that the relative locations and orientations of each stroke are preserved in their written characters. The second assumption motivated us to develop an effective recognition algorithm that relies on the relative stroke locations on each orientation and is invariant of the character's internal distribution.

(a) Proper template of the Chinese character for "five".

(b) An example of a handwritten Chinese character for "five" with improper stroke positions.

Figure 3: A comparison of a correct template and an input with improper stroke positions.

Figure 3 shows an example of a handwriting input for the character "five" with improperly-positioned strokes. The middle horizontal stroke segment is located too close to the top instead of being centered, and the vertical stroke segment on the right is also located very off-centered. Euclidean distance-based template-matching algorithms [11, 30] would not perform well on these handwriting inputs because: 1) the improper distribution of the stroke points can introduce errors in matching the template points to the corresponding input points [30], and 2) the calculated distances between the template and candidate point clouds could be very large and lead to lower confidence of the template corresponding to the candidate in the recognition process [1, 2, 31].

In order to improve the tolerance for imbalanced stroke distributions, we applied Dynamic Time Warping (DTW) [4] for input symbol recognition. DTW is an efficient algorithm for matching two sequences with similar patterns but different lengths or paces. The algorithm's dynamic programming nature makes it possible to find an optimal match between such two sequences. As observed from the novice students' written input, the characters have projection arrays with similar patterns but distinct internal intervals to the templates. We therefore applied DTW on the four orientations for template-matching. The final template-matching distance is a weighted sum of the DTW costs of these four orientations.

### 3.3 Technique Mistake Detection

While it is not a challenging task for novice CSL students to write characters that are visually similar to the templates, these students can make numerous written technique mistakes. In order to better guide these students, it is important that our techniques accurately detect their mistakes and provide the appropriate corresponding feedback. The following sections describe how our proposed sketch recognition techniques address mistakes for written techniques that are conventionally taught in Chinese character instruction [32]: stroke count, stroke order, and stroke directions.

#### 3.3.1 Stroke Count

We observed that Chinese characters can be formed as a set of polyline strokes connecting or intersecting with each other. A complex combination of these polyline strokes can be challenging for novice students, such as determining how to separate these strokes. We anticipated four possible type of mistakes that can occur from proper stroke count: concatenated, broken, missing, and extra strokes (Table 2). The first example character contains three strokes, where the first stroke is written horizontally from left to right, and then extends to another left-diagonal segment; and the second stroke starts from the end point of the first one, then goes from top to bottom, and finally ends with a hook towards the left. This character is however often mistakenly written with only two strokes, leaving the first and second stroke concatenated due to their respective starting and ending points being in the same location, as shown in the corresponding example. Strokes with multiple line segments can confuse users on how they should break these strokes. In the broken strokes example, the character has a stroke that contains two line segments, while the user could potentially write them separately. The third example shows a more complicated character with two vertical strokes contained inside a square, but with one of these strokes missing. The fourth example shows two visually similar but completely distinct characters, where adding a short stroke to the bottom of the template character will produce a different character altogether.

In order to detect specific mistakes from the user's input and provide subsequent feedback, we required an efficient algorithm for matching strokes from the template and input characters. We propose an approach to find the stroke correspondences for the following conditions.

When the input character has same amount of strokes as the template character, our approach identifies the student's written input as satisfying the correct stroke count. Therefore, the task is to

Table 2: Four types of stroke count mistakes from novice CSL students.

| Mistake Type | Example | Template |
|---|---|---|
| Concatenated strokes |  |  |
| Broken strokes |  |  |
| Missing strokes |  |  |
| Extra strokes |  |  |

locate the one-to-one correspondence of each input stroke to each template stroke for subsequently assessing for proper stroke order and direction. We applied a greedy algorithm for finding matching strokes that approximate closely to optimal results due to the low number strokes for each character. The algorithm works as follows: for each stroke in the input $s_i$, locate an unmatched stroke $t_j$ in the template to minimize the Hausdorff distance between $s_i$ and $t_j$. This algorithm creates the one-to-one correspondences between the input and template strokes.

When the input character contains fewer strokes than the template, our approach either identifies that the concatenated strokes exist in the input or that some template strokes are missing. We then propose a greedy algorithm that relies on Hausdorff distances and directed Hausdorff distances to find the stroke correspondence. As introduced in [11], the Hausdorff distance between two point sets $A$ and $B$ is defined as:

$$H(A,B) = \max(h(A,B), h(B,A)) \tag{1}$$

where

$$h(A,B) = \max_{a \in A}(\min_{b \in B}||a-b||) \tag{2}$$

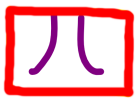
is defined as the directed Hausdorff distance from $A$ to $B$. The directed Hausdorff distance denotes the maximum distance from each point in $A$ to its closest point in $B$. From Equation 2, the directed Hausdorff distance measures how $A$ is visually similar to a subset of $B$. The value $h(A,B)$ can be very small when $A$ approximately overlaps a subset of $B$, while the $H(A,B)$ distance can be very sensitive to outliers. Observing these characteristics, we propose a greedy algorithm to find a one-to-many stroke correspondence from the input character to template character by iterating through each input stroke and locating the most optimal set of matching template strokes. The algorithm works as follows:

1. For each input stroke, create an empty list that stores the matching template strokes, and sort all template strokes that have not been matched to any input stroke by their directed Hausdorff distances to the input stroke.
2. Gradually add template strokes to the list in the order they are sorted until the Hausdorff distance between the list and the input stroke increases.
3. The final lists are the result of the stroke matching step.

When the input character has more strokes than the template character, our approach identifies that it must either be due to a broken stroke or to extra strokes in the input. For this situation, we can apply the above algorithm to find a one-to-many correspondence from the template strokes to input strokes, similarly to the many-to-one correspondence from the input strokes to template strokes.

In addition to providing binary feedback for indicating whether the user has written the correct stroke counts, our approach also allows for interfaces to provide specific interactive instructions to instruct users. For example, if there are extra or missing strokes detected, our demonstration interface uses the output from our approach to highlight the extra or missing strokes in the written input on the writing area. When broken strokes are detected, the interface highlights the separated stroke segments with different colors to indicate that they should be written in one stroke. When there are concatenated stroke mistakes, the interface highlights the specific strokes that should be written with multiple strokes Figure 4.

(a) Feedback for broken strokes



(b) Feedback for concatenated strokes

Figure 4: Feedback for the broken and concatenated strokes: (a) the character "five" should be written with 4 strokes, where second stroke is the middle stroke with one horizontal and one vertical line segment, (b) the rectangle that surrounds the character "four" should be written with three strokes.

### 3.3.2 Stroke Order

Stroke order in students' handwriting inputs is classified as correct if each stroke in the input is written in the same chronological order as in the template. For inputs that have correct stroke counts, the approach reviews the one-to-one stroke correspondence and identifies the stroke order as correct only if each element in the list is equal to its index. For inputs that have more strokes than its corresponding template, the approach reviews the stroke order as correct only if both the correctly-written strokes and the broken strokes are written in correct chronological order. If a template stroke is detected with multiple broken strokes, we take each corresponding input stroke $s_i$ and locate its matching part in the template by locating points $P_{si}$ and $P_{ei}$ on the template stroke, which has the minimum distance from the start and end points to that input stroke, respectively. The broken strokes are in the correct order only if for each $i$, $P_{si}$ has appeared earlier than $P_{s(i+1)}$ in the template. Moreover, the approach also requires that for each template stroke $t_j$, the corresponding input stroke must all appear earlier than all input strokes that corresponds to $t_{(j+1)}$, so as to be considered to have correct stroke order. The stroke order assessment for characters that have fewer strokes than the template works similarly. To give feedback to users in the demonstration interface, we highlight the strokes in the input that are not in the correct chronological order, so as to let users know which strokes are ordered incorrectly. Figure 5a shows an example of the demonstration interface's feedback on written input with incorrect stroke order.

### 3.3.3 Stroke Direction

Stroke direction is defined as the chronological order of the start point and end point of each stroke. To determine if each stroke is written in the correct direction, our approach calculates a vector $\vec{v_s}$ from the start to the end point and a vector $\vec{v_t}$ based on either a complete or a part of a template stroke that it corresponds to, and calculates $\cos(\vec{v_s}, \vec{v_t})$ as the cosine between these two vectors. A

stroke is considered to have the correct direction only if the cosine value is positive. For each input stroke that has incorrect direction, the demonstration interface receives feedback from our approach by showing a green dot moving from the correct start point to the correct end point of the given stroke on the canvas to indicate the correct stroke direction. The visualization feedback for stroke direction mistakes is shown in Figure 4, which shows the interface's feedback on a written character with a stroke direction mistake.

(a) Feedback for wrong stroke order.

(b) Feedback for wrong stroke direction

Figure 5: Feedback from the demonstration interface for stroke order and direction mistakes.

### 3.4 Visual Quality Assessment

The goal of this part of our work aims at finding out important features that reflect how a user becomes proficient in writing Chinese characters. Our approach examines both stroke and vision level features that reflect not only how the input is similar to the template as a whole, but also the internal balance of the character. It also take into account writing speed, which is significant to indicate how familiar and accurate the user is with writing the characters. After feature extraction, we apply feature selection methods [33] to select an optimal subset of features that is most descriptive on the visual quality of the user's written inputs. The aim is to further apply these selected features to improve assessment from our approach, so that interfaces can potentially generate automatic grades and feedback for the written input's visual structures. Table 3 lists the features that we extracted for each written input.

## 4 Results and Discussion

### 4.1 Evaluation

To evaluate the performance of our recognition techniques, we collected data from three groups of students at a large public research university. Group 1 consisted of 11 students with native written Chinese language fluency and who have used Chinese as their primary language since childhood. These students were asked to write each of the 27 characters—three times each—to the best of their abilities. This collected data was used to evaluate our symbol recognition algorithm and as labeled data for training and testing the machine learning algorithm that classified between proper and improper handwriting inputs. Group 2 consisted of 9 students who each have more than ten years of experience writing Chinese characters. This group was asked to provide characters with casual writing styles. We used data from this second group to evaluate our stroke matching algorithm and to compare the results with similar East Asian language educational interfaces [26]. Group 3 consisted of 10 novice language students who have little to no experience with writing Chinese. These users were asked to write each character—three times each—to mimic the template character presented to them, and use the feedback provided our techniques via the demonstration interface to improve their writing techniques. Data from group 3 was used to evaluate our recognition algorithms on both visual structure written technique, and labeled as improperly drawn data to train and test the machine learning system for classification. Novice users were also asked to share their thoughts and provide remarks on the performance of our recognition techniques through a survey.

Table 3: Extracted features from the written input.

| Feature | Description |
|---|---|
| Bounding box ratio | centroid location ($y$-axis) |
| Centroid location ($x$-axis) | horizontal distance between input's centroid and center compared to template [23] |
| Centroid location ($y$-axis) | vertical distance between input's centroid and center compared to template [23] |
| Hausdorff distance | distance based on Hausdorff metric [11] |
| Tanimoto coefficient | first similarity measurement of binary image patterns [11] |
| Yule coefficient | second similarity measurement of binary image patterns [11] |
| Stroke length distribution | difference in distribution strokes's lengths within given character |
| Average stroke orientation similarity | average cosine value of angles formed by the start-to-end vector of each stroke in the template and its corresponding stroke in the input |
| Minimum stroke orientation similarity | minimum cosine value of angles formed by start-to-end vector of each stroke in the template and its corresponding stroke in the input |
| Average speeds | sum of average speeds |
| Speed fluidity | ratio of minimum and maximum speed that user writes the character |
| Horizontal projection difference | difference in horizontal projection |
| Vertical projection difference | difference in vertical projection |

We evaluate the performance of our recognition techniques on four different criteria. First, we evaluated the proposed techniques against state-of-the-art template matching algorithms and techniques from previous East Asian language educational interfaces [26, 30], respectively. We conducted this evaluation using data from Groups 1 and 3. Second, we evaluated our techniques for assessing students' written techniques. For this criteria, we evaluated the stroke matching algorithm on data from both groups 2 and 3, and then evaluate other written technique assessments on data from only group 3. Third, we evaluated how well the features performed for classifying between proper and improper character writing using data from both groups 1 and 3. Lastly, we evaluated the usability the feedback of our techniques via the demonstration interface based on the technique mistakes that novice users made over time and their feedback based on the demonstration interface using data from group 3.

### 4.2 Visual Structure Recognition

We compare the recognition results and rankings of the correct answers of our proposed sketch recognition techniques with the following methods:

1. **$P**: a state-of-the-art template matching algorithm for multi-stroke gestures based on point clouds [30]
2. **BopoNoto**: a sketch-based educational interface used for Mandarin phonetic symbols writing practice [26]

All three algorithms were run on the datasets from both the expert users (group 1) and novice users (group 3). As seen in Figure 6, our proposed techniques achievged 98% and 85% recognition rates for expert and novice users, respectively, which exceeds BopoNoto's 94% and 79%, and $P's 89% and 73%.

Each template matcher returns a list of labels of their similarity to the input, where we consider template matchers to have better performance for more correct templates that are listed in the top several results in the list. For examining our proposed techniques,

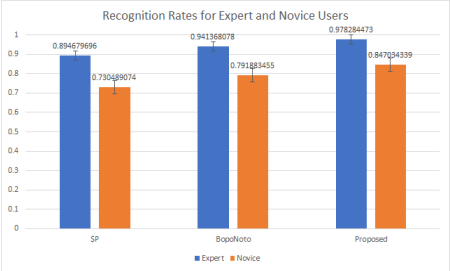



Figure 6: Recognition rates on expert and novice data.

we classify each input as correct if it ranks in the top three results of the recognition list. Figure 7 shows our proposed recognition techniques as achieving 100% and 95% in this metric for expert and novice data, respectively, exceeding BopoNoto's 96% and 83%, and $P's 97% and 90%.

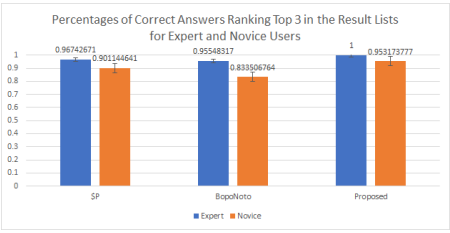



Figure 7: Percentage of written inputs whose corresponding templates rank top three in the result.

### 4.3 Written Technique Assessment

We evaluate our proposed recognition techniques' ability to assess users' writing techniques and accuracy in detecting their written technique mistakes.

#### 4.3.1 Stroke Matching

We tested our proposed stroke matching algorithm on both novice data (group 3) and cursive writing samples from 9 expert users (group 2). For samples that have correct stroke counts, we compared our method with BopoNoto [26], which uses a similar greedy algorithm for locating stroke correspondences. The results show that our algorithm can locate correct stroke correspondences with an accuracy of 98.5%, which exceeds BopoNoto's 92.4%. While BopoNoto is not able to deal with characters with concatenated and broken strokes, our algorithm can correctly locate stroke correspondences for samples that have less strokes than the template with an accuracy of 85.6%, and for those that have more strokes with an accuracy of 97.3%.

#### 4.3.2 Stroke Order Assessment

In our demonstration interface, we provided binary feedback from our proposed techniques to students regarding the stroke order correctness of each of their written characters. BopoNoto [26] proposed an effective way to determine stroke order for characters with correct stroke counts, but fails when the stroke count is incorrect. Observing that more information is yet to be retrieved and utilized in broken strokes and concatenated strokes, our proposed method aims at providing richer feedback by assessing and providing feedback on all characters regardless of stroke count correctness.

We first evaluated the accuracy of our stroke order assessment method on characters with correct stroke counts, and then compared it against BopoNoto [26]. The results showed that our method achieves an accuracy of 98.6% and an f-measure of 99.1%, performing slightly better than BopoNoto's 96.2% and 95.4% (Table 5).

Table 4: Confusion matrix of the proposed stroke order assessment for correct stroke count.

| | **Predicted: correct** | **Predicted: incorrect** |
|---|---|---|
| **Actual: correct** | 1143 | 17 |
| **Actual: incorrect** | 4 | 287 |

Table 5: Confusion matrix of BopoNoto's stroke order assessment for correct stroke count.

| | **Predicted: correct** | **Predicted: incorrect** |
|---|---|---|
| **Actual: correct** | 1079 | 83 |
| **Actual: incorrect** | 12 | 277 |

We then evaluated our proposed techniques' ability to assess stroke order on characters with incorrect stroke counts. The results show that our method was able to accurately judge stroke order when stroke counts are incorrect with an accuracy of 87.8% and an f-measure of 89% (Table 6).

Table 6: Confusion matrix of the proposed stroke order assessment for incorrect stroke count.

| | **Predicted: correct** | **Predicted: incorrect** |
|---|---|---|
| **Actual: correct** | 102 | 25 |
| **Actual: incorrect** | 0 | 49 |

#### 4.3.3 Stroke Direction Assessment

We evaluated the binary classification result on the correctness of stroke directions on characters with correct and incorrect stroke counts separately. We first test our algorithm on users' inputs with correct stroke counts. Table 7 shows our method accurately classifing between characters with and without direction errors with an accuracy of 98.3% and an f-measure of 99.1%, which greatly exceeds BopoNoto's 80.4% and 88.1%. Table 9 shows our assessment method achieves an accuracy of 87.2% and an f-measure of 91.9% on the detection of incorrect directions for incorrect stroke count.

Table 7: Confusion matrix of the proposed stroke direction assessment for correct stroke count.

| | **Predicted: correct** | **Predicted: incorrect** |
|---|---|---|
| **Actual: correct** | 1317 | 20 |
| **Actual: incorrect** | 5 | 106 |

Table 8: Confusion matrix of BopoNoto's stroke direction assessment for correct stroke count.

| | **Predicted: correct** | **Predicted: incorrect** |
|---|---|---|
| **Actual: correct** | 1055 | 282 |
| **Actual: incorrect** | 2 | 109 |

Table 9: Confusion matrix of the proposed stroke direction assessment for incorrect stroke count.

| | **Predicted: correct** | **Predicted: incorrect** |
|---|---|---|
| **Actual: correct** | 130 | 23 |
| **Actual: incorrect** | 0 | 27 |

### 4.4 Writing Quality Assessment

In this section, we evaluated the importance of features on distinguishing between properly- and improperly-written characters. We used data from groups 1 and 3 as training data, which were automatically labeled as "good" and "bad" writing inputs, respectively. We performed a Best-First Search algorithm [33] to select the subset of features that have the best separability between the two labeled datasets. The feature subset selection result showed that 10 out of the 13 features were selected, excluding $f_6$ (Yule coefficient), $f_9$ (Minimum stroke orientation similarity), and $f_{13}$ (Vertical projection difference). Based on the features selected, we applied the Random Forest algorithm [15] on our dataset using selected features to classify between expert and novice data and leave-one-user-out cross-validation. The results achieved a weighted f-measure of 0.804, as shown in Table 10.

Table 10: Detailed accuracy by class with leave-one-user-out cross-validation.

| **Class** | **Precision** | **Recall** | **F-measure** |
|---|---|---|---|
| Expert | 1 | 0.701 | 0.798 |
| Novice | 1 | 0.698 | 0.811 |
| Overall | 1 | 0.699 | 0.804 |

### 4.5 Usability

In this study, we evaluated the usability of our proposed recognition techniques from the feedback displayed in the demonstration user interface. The goal was to analyze how the feedback to CSL students had helped improve their character writing. Users from group 3 were asked to write each character by mimicking the template provided, and use the feedback provided by the demonstration interface to improve their subsequent attempts. We then counted the number of written technique mistakes for each attempt, and observed whether the feedback from the demonstration interface was able to assist them in better character writing. After collecting their writing data, we asked the novice users to give their review of the feedback from our demonstration interface by scoring between 1 and 5—where 5 was the highest—to indicate how each part of the feedback had helped them improve their character writing. We first analyzed the data on the number of technical mistakes that users made on each attempt. Figure 8 shows the number of mistakes from the users' handwriting greatly decreased when using feedback from the demonstration interface.

Figure 9 shows the scores each user gives to our technique feedback. The users' average ratings for stroke count, order, and direction feedback were 4.22, 3.33, and 4.11, respectively. We observed that users were very satisfied on the interface's feedback on stroke count and stroke direction, but were sometimes confused on the feedback for stroke order. We hope to discover an improved way for providing stroke order feedback in the near future. Users also mentioned that the way we highlighted broken and concatenated strokes was very helpful to inform them on how to separate the strokes. One user mentioned that the displayed animated trace on strokes with incorrect directions helped them more clearly remember which stroke should go in which direction. Some users also mention

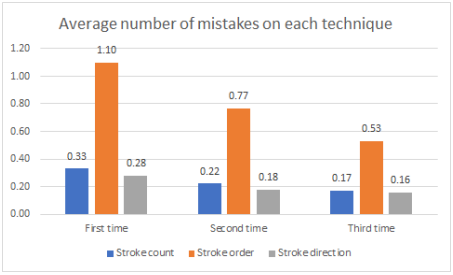



Figure 8: Average number of mistakes on each attempt for each technique.

that it was not as clear to understand how they should correct their stroke order mistakes based on the feedback. These suggestions were helpful as considerations for specifying what educational interface designers should consider when utilizing our proposed sketch recognition techniques.

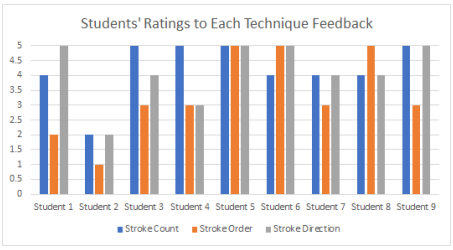



Figure 9: Students' ratings to each technique feedback.

### 4.6 Further Considerations

From our proposed recognition techniques, we focused on testing with 27 representative Chinese characters to demonstrate and evaluate our technique's performance. However, additional lesson plans require a much larger set of characters and may challenge our recognition techniques as the number of characters increases. One possible solution would involve creating smaller character subsets that correspond to individual book chapters or lessons, so that the recognition techniques can more easily scale. Another possible solution includes incorporating additional heuristic rules for further distinguishing between more similar characters.

Another consideration is that we assume users only write single characters. However, instructors may be interested in more sophisticated interfaces with multi-character inputs to reflect larger possible vocabularies, which violates our assumptions. This problem was similarly considered in [26], and the work proposed segmenting multiple symbols with naive block spacing assumptions to accommodate multiple symbol writing input. We believe that this approach has potential to address our assumption's limitations for more sophisticated multi-character writing input.

## 5 Conclusion and Future Work

In this work, we presented nove sketch recognition techniques for providing richer assessment of novice students' writing of Chinese characters. Compared to techniques in prior character writing ITS interfaces, the assessment performance of our proposed techniques better performed in assessing students' written characters. We developed a new method for handwriting recognition that achieved an accuracy of 86% on novice learners' data. Our technique assessment system accurately detected students' mistakes on stroke count, stroke order, and stroke direction, and can enable interfaces to provide real-time feedback for assisting students in improve their

future writing performance. We also collected written character inputs from both experts and novice students, and discovered 10 features that performed well for writing quality assessment. We used these features to classify expert and novice users' written character data using machine learning techniques, and achieved an f-measure of approximately 80%. We evaluated the usability of the feedback from our proposed techniques on a demonstration interface for novice CSL students, and observed that users had overall positive feedback of the feedback.

One potential follow-up work includes considering pressure as a potential feature for Chinese character writing, since we believe it may be correlated to speed and can distinguish between novice and experienced writers. We would also like to consider integrating visual quality assessment into our proposed recognition techniques by developing an automatic grading algorithm based on features that we observed to perform well, so that users may not only learn how to write each character visually and technically correct, but to also learn how to achieve decent visual quality. Furthermore, we are interested in developing a fully-functional educational interface that employed our proposed recognition techniques at an introductory Chinese language course for assessing its performance and how well it complements human instructors' existing pedagogical practices.

## REFERENCES


[1] L. Anthony and J. O. Wobbrock. $ n-protractor: A fast and accurate multistroke recognizer. In *Proc. GI '12*, pp. 117–120. Canadian Information Processing Society.

[2] L. Anthony and J. O. Wobbrock. A lightweight multistroke recognizer for user interface prototypes. In *Proc. GI '10*, pp. 245–252. Canadian Information Processing Society.

[3] Z.-L. Bai and Q. Huo. A study on the use of 8-directional features for online handwritten chinese character recognition. In *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*, pp. 262–266. IEEE.

[4] D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop '94*, pp. 359–370. Seattle, WA.

[5] M. E. Everson. Literacy development in chinese as a foreign language. *Teaching Chinese as a foreign language: Theories and applications*, pp. 97–111, 2009.

[6] T. Hammond and R. Davis. Ladder, a sketching language for user interface developers. *Computers & Graphics*, 29(4):518–532, 2005.

[7] C.-C. Han, C.-H. Chou, and C.-S. Wu. An interactive grading and learning system for chinese calligraphy. *Machine Vision and Applications*, 19(1):43–55, 2008.

[8] J. Herold and T. Stahovich. Classyseg: A machine learning approach to automatic stroke segmentation. In *Proc. SBIM '11*, pp. 109–116. ACM, New York, NY, USA.

[9] Z. Hu, Y. Xu, L. Huang, and H. Leung. A chinese handwriting education system with automatic error detection. *Journal of Software*, 4(2):101–107, 2009.

[10] Q. Huo, Y. Ge, and Z.-D. Feng. High performance chinese ocr based on gabor features, discriminative feature extraction and model training. In *Proc. ICASSP '01*, vol. 3, pp. 1517–1520. IEEE.

[11] L. B. Kara and T. F. Stahovich. An image-based, trainable symbol recognizer for hand-drawn sketches. *Computers & Graphics*, 29(4):501–517, 2005.

[12] H.-h. Kim, P. Taele, S. Valentine, E. McTigue, and T. Hammond. Kimchi: A sketch-based developmental skill classifier to enhance pen-driven educational interfaces for children. In *Proc. SBIM '13*, pp. 33–42. ACM, New York, NY, USA.

[13] J. J. LaViola, Jr. and R. C. Zeleznik. Mathpad2: A system for the creation and exploration of mathematical sketches. *ACM Transactions on Graphics*, 23(3):432–440, aug 2004.

[14] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[15] A. Liaw, M. Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[16] R. Romero, R. Berger, R. Thibadeau, and D. Touretzky. Neural network classifiers for optical chinese character recognition. In *Proceedings of the Fourth Annual Symposium on Document Analysis and Information Retrieval*, pp. 385–98, 1995.

[17] R. D. Romero, D. S. Touretzky, and R. H. Thibadeau. Optical chinese character recognition using probabilistic neural networks. *Pattern recognition*, 30(8):1279–1292, 1997.

[18] D. Rubine. Specifying gestures by example. In *Proc. the 18th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '91, pp. 329–337. ACM, New York, NY, USA, 1991.

[19] N. Shahzad, B. Paulson, and T. Hammond. Urdu qaeda: Recognition system for isolated urdu characters. In *Proc. IUI SR '09*, pp. 1–5.

[20] T. F. Stahovich and H. Lin. Enabling data mining of handwritten coursework. *Computers & Graphics*, 57(31–45), jun 2016.

[21] T. F. Stahovich, E. J. Peterson, and H. Lin. An efficient, classification-based approach for grouping pen strokes into objects. *Computers & Graphics*, 42:14–30, aug 2014.

[22] Y.-M. Su and J.-F. Wang. A novel stroke extraction method for chinese characters using gabor filters. *Pattern Recognition*, 36(3):635–647, 2003.

[23] R. Sun, Z. Lian, Y. Tang, and J. Xiao. Aesthetic visual quality evaluation of chinese handwritings. In *IJCAI '15*, pp. 2510–2516.

[24] P. Taele. Freehand sketch recognition for computer-assisted language learning of written east asian languages. MS Master's Thesis, Texas A&M University (TAMU), College Station, TX, USA, December 2010. Advisor: Tracy Hammond. 96 pages.

[25] P. Taele, L. Barreto, and H. Tracy. Maestoso: An intelligent educational sketching tool for learning music theory. In *Proc. IAAI '15*, pp. 3999–4005. AAAI, Palo Alto, CA, USA.

[26] P. Taele and T. Hammond. Boponoto: An intelligent sketch education application for learning zhuyin phonetic script. In *Proc. DMS '15*, pp. 101–107.

[27] P. Taele and T. Hammond. Hashigo: A next-generation sketch interactive system for japanese kanji. In *Proc. IAAI '09*, vol. 9, pp. 153–158.

[28] P. Taele and T. Hammond. Lamps: A sketch recognition-based teaching tool for mandarin phonetic symbols i. *Journal of Visual Languages & Computing*, 21(2):109–120, 2010.

[29] L. Tsung and K. Cruickshank. *Teaching and learning Chinese in global contexts: CFL worldwide*. Bloomsbury Publishing, 2010.

[30] R.-D. Vatavu, L. Anthony, and J. O. Wobbrock. Gestures as point clouds: a $ p recognizer for user interface prototypes. In *Proc. ICMI '12*, pp. 273–280. ACM.

[31] J. O. Wobbrock, A. D. Wilson, and Y. Li. Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes. In *Proc. the 20th annual ACM symposium on User interface software and technology*, pp. 159–168. ACM, 2007.

[32] J. Z. Xing. *Teaching and learning Chinese as a foreign language: A pedagogical grammar*, vol. 1. Hong Kong University Press, 2006.

[33] L. Xu, P. Yan, and T. Chang. Best first strategy for feature selection. In *ICPR '88*, pp. 706–708. IEEE.