Model-Based Annotation of Online Handwritten Datasets

Anand Kumar, A. Balasubramanian, Anoop Namboodiri, C.V. Jawahar

Model-Based Annotation of Online Handwritten Datasets

Anand Kumar, A. Balasubramanian, Anoop Namboodiri and C.V. Jawahar
Center for Visual Information Technology,
International Institute of Information Technology,
Gachibowli, Hyderabad - 500 032, INDIA
jawahar@iiit.ac.in

Abstract

Annotated datasets of handwriting are a prerequisite to attempt a variety of problems such as building recognizers, developing writer identification algorithms, etc. However, the annotation of large datasets is a tedious and expensive process, especially at the character or stroke level. In this paper we propose a novel, automated method for annotation at the character level, given a parallel corpus of online handwritten data and the corresponding text. The method employs a model-based handwriting synthesis unit to map the two corpora to the same space and the annotation is propagated to the word level and then to the individual characters using elastic matching. The initial results of annotation are used to improve the handwriting synthesis model for the user under consideration, which in turn refines the annotation. The method can take care of errors in the handwriting such as spurious and missing strokes or characters. The output is stored in the UPX-InkML format.

Keywords: Annotation, Synthesis model, Elastic matching, Dynamic programming.

1. Introduction

Annotated datasets of handwriting covering a variety of writing styles are essential for the development and evaluation of handwriting recognition engines, especially those which utilizes the data-driven learning approaches [1, 2, 3, 4, 5]. Lack of linguistic resources for many scripts, in the form of annotated handwriting datasets has been one of the major hurdles in building recognizers for them. Annotation of such large corpora of handwritten data is a time-consuming and error-prone process, especially at the character level. On the other hand, plain transcripts of handwritten data may be available in many cases, such as when the handwriting is based on an existing text. The process of data collection and annotation for handwritten text could be carried out in a variety of settings. Each has its own assumptions:

Unrestricted Data: Unrestricted data is collected in the natural settings of handwriting (say meeting notes) and is ideally suited for building real-world systems. Often there is no restriction on the type of data, and the annotation has to be done manually or with the help of a semi-automatic labeling tool. Such an approach was presented in [6]. Their data set consisted of around nine hundred sheets of cursive writing. The segmentation was done based on the word spacing and annotation was carried out at word level. Our method annotates the words at character level.

Designed Text: To make the annotation simple and comprehensive, one often resorts to writing of pre-defined text, whose contents are tailored to the needs of the problem that is attempted. Usually the same text is collected from multiple writers.

Dictation: This is a variant of the previous model, where the text is dictated to a group of writers, who will all write the same text. The content could also be non-designed as in the case of lecture notes. In both cases, as the number of writers increases, manual labeling becomes difficult.

Data Generation: An alternate way to obtain handwritten data is to develop a model of the handwriting process and use it for automatic synthesis of handwritten data. This method is appealing for applications requiring large quantities of data since the generated data will already be annotated. However, the quality of the synthesized data will be limited by the accuracy of the handwriting model.

In each of the first three data collection methodologies, one can have a parallel corpus of text due to two factors: i) one could hire an experienced typist in a language to generate transcripts of available handwritten data, and ii) many handwritten datasets are collected based on text that is already available in the electronic form. However, such a text need not exactly align with the handwriting due to errors in the handwriting or the transcription process. In this paper we propose a model-based annotation framework, where the handwriting style of the writer is learnt and used to propagate the transcription to words and characters. The method automatically aligns the handwritten data with the textual data and thus generate annotation of handwriting at character level.

The problem of propagation might seem relatively simple to solve since we have parallel corpora in the handwritten and text formats. However, this is true only if: i) we have an error free segmentation of the handwritten data at the word and character levels, and ii) the transcription strictly matches the handwriting, without any errors. Both these assumptions are often violated in real world handwritten datasets and their transcriptions. Hence the process of alignment of the handwritten data with a corresponding corpus includes the identification of the word

and character boundaries of the handwritten data as well as the mapping of the handwritten strokes to the characters in the text.

The problem of annotation is not always restricted to labeling of the content (or text) information. Additional details such as the language or script, particulars of the writer, writing conditions, etc. may have to be added for certain applications. In this paper, we restrict ourselves to the annotation of content of handwritten data. A related problem is that of storage of the annotation in a format that allows for incremental efficient and easily accessible labeling [7].

2. Overview of Previous Work

The problem of alignment of parallel handwritten and text corpora has been addressed before in the context of segmentation of text lines to words by Zimmerman and Bunke [2]. Their method assumes the presence of a reliable recognition engine, building resources for which is the goal of our annotation process. Tomai *et al.* [3] proposes a similar approach to annotate words of historic handwritten data, where the recognizer is constrained to output the words in the transcript.

The work that is closest to our approach is by Kornfield *et al.* [4], where word images are matched to text based on global properties of the word extracted from both the handwritten word images as well as text words that are rendered using a specific font. The matching is done using dynamic time warping (DTW). Rothfeder *et al.* [5] extends this approach to use an HMM for matching the words, that handles some amount of errors in the segmentation as well as transcription process.

The problem of representation of the annotation or ground truth has been studied before. Guyon *et al.* [8] presented UNIPEN standard for representing annotated datasets of online handwritten data. Bhaskarbhatla *et al.* [1] presented an XML-based representation scheme for annotation of online handwritten data. A tool was created for annotation based on the proposed representation. Although the tool attempts to address the requirements of creation of annotated data sets of handwritten data in different scripts, the process still requires selection and annotation of individual characters.

In this paper, we propose a model-based synthesis and annotation framework, that automatically segments and aligns transcripts for online handwritten documents. In this sense, our approach combines the advantages of the segmentation algorithm in [2] and the alignment capabilities of [4]. Moreover, we do not assume the availability of a recognizer and the algorithm is robust towards noise in the handwriting, as well as errors in the transcription process. The framework allows us to learn and update the handwriting model for a writer from the initial annotated data, which in turn can improve the annotation. This approach can produce annotations at the character level for documents in Indian languages.

3. Proposed Algorithm

The overall approach is illustrated in Figure 1. The input to the algorithm is a sequence of online handwritten data referred to as the input handwriting and the corresponding transcription, referred to as the text sequence. A synthesis module is used to convert the text sequence to the corresponding handwritten form using a model of handwriting of the writer. This forms the reference handwriting, where the segmentation and ground truth are accurately known. The input handwriting is then matched with the reference handwriting using a two-stage elastic matching module. After the best match is identified, the ground truth is propagated to the words and characters of the input handwriting. The handwriting synthesis module can refine the parameters of the handwriting model of the writer and repeat the annotation process using the updated handwriting synthesizer and the available annotation. Thus the framework allows us to refine the annotation information over time. We will now describe the process formally.

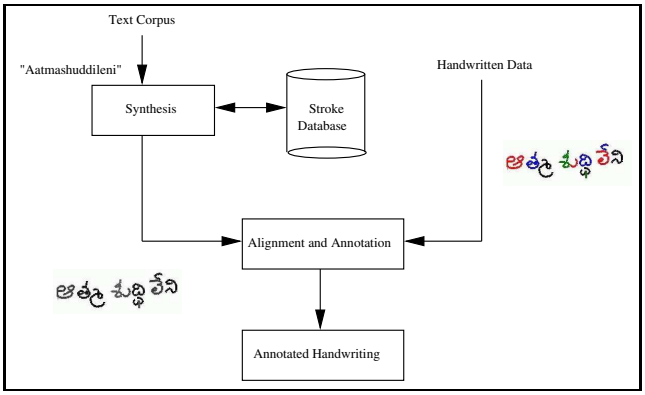

Figure 1. Block diagram of the annotation process.

3.1. Definitions

Online handwritten data is represented as a sequence of strokes: $\langle s_1, s_2, \cdots, s_K \rangle$, where a stroke is defined as the trace of the samples starting from a pen-down to the following pen-up. A word, w_i, forms a contiguous sub-sequence, $\langle s_j, s_{j+1}, \cdots, s_{j+k} \rangle$, where s_j is the starting stroke of the word w_i. Hence the data may also be represented as a sequence of words: $W = \langle w_1, w_2, \cdots, w_N \rangle$. The text corpus that corresponds to the handwritten data is a sequence of characters, that are separated by a blank character at the word boundaries: $T = \langle t_1, bl, t_2, bl, \cdots, t_M \rangle$. Each t_i corresponds to a text word, which is a sequence of characters: $\langle c_1, c_2, \cdots, c_J \rangle$.

The input handwriting, is a sequence of strokes, which could be grouped into a sequence of words. The problem of segmentation or grouping of strokes into words is often difficult because there might not be a clear spatial separation between the last stroke of a word and the first stroke of the next. We utilize the blank character information present in the text data to aid the word segmentation process. During the segmentation process, the stroke sequences are also aligned with the characters in the text,

thus generating a word-level and character-level annotation of the handwritten data.

An important assumption in our approach is that a character is always composed of an integral number of strokes. Thus the problem of character level annotation is reduced to finding out the set of strokes that correspond to a particular character in the text representation. The approach could also be extended to work with cursive writing, where a single stroke might span multiple characters.

A primary requirement to do alignment, either at the word level or at the character level is that there is a mechanism to match a text word/character with handwritten strokes. However such a matching is not trivial due to the variations that are possible within a single character class in the handwritten data.

3.2. Handwriting Synthesis

The process of generating a handwritten equivalent of a given text word is referred to as handwriting synthesis. Depending on the application, the synthesis could generate a handwritten word in the writing style of a specific user or a generic one. Our approach to synthesis of handwritten words is as follows.

The synthesis module consists of three parts: i) Conversion of a text word into a sequence of handwritten stroke classes, ii) Computation of candidate strokes that could be used for each stroke class, and iii) Computing the spatial layout of the candidate strokes to arrive at the final handwritten word.

The set of stroke classes that constitute a word or character is learnt from a corpus of training samples that are annotated at the character levels. The learning could be writer specific or writer independent as the application mandates. Template strokes are also identified for each stroke classes in this process. To synthesize the strokes for a particular word, we use a deformation model that takes the template strokes and modifies them within the parameters of the desired writing style. A third step learns the spatial bi-gram distribution of the strokes from the training corpus. This is used to combine the synthesized handwritten strokes into a single word. Details of the handwriting synthesis algorithm used in this work can be found in [9]. Any other synthesis algorithm can also be used in place of this. Our current synthesis procedure fits well for Indian language scripts, in which the experiments are carried out. Figure 2 shows a sample word in *Telugu*, that is converted to handwriting using our approach. Once the handwritten word is generated, the strokes are mapped to a feature space representation for matching.

3.3. Matching Handwritten Strokes

Distance or dissimilarity between two scribbles (sets of strokes) is computed using a set of features extracted from the group of strokes. Each stroke consists of a sequence of sample points, (x, y) that describes the trace of the pen during writing. The strokes are first converted into a sequence of feature vectors, extracted from each of the

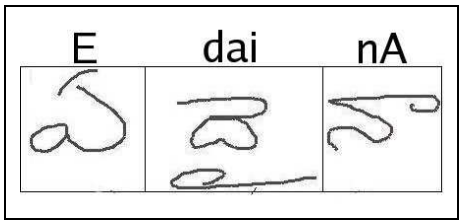

Figure 2. Figure illustrating synthesis. A sample word "EdainA" in Telugu language (shown in ITRANS) and the corresponding synthesized word.

sample points. The feature vector consists of:

1. The direction, θ, of the tangent to the stroke curve
2. The curvature, c, of the stroke at the sample point, and
3. The height, h, of the sample point from the word baseline

Figure 3 illustrates the definition of the three features. The left figure shows the height, the middle one shows the direction and the last figure shows the curvature features of a handwritten character.

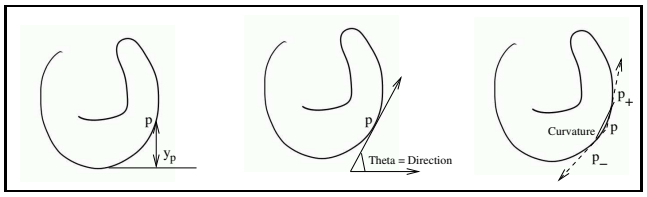

Figure 3. Figure showing the three features: The height, direction and the curvature extracted from the sample points on the curves.

The distance between two feature vectors $F_1 = \langle \theta_1, c_1, h_1 \rangle$ and $F_2 = \langle \theta_2, c_2, h_2 \rangle$ is defined as the weighted Euclidean distance between the two vectors:

$$D^2 = k_\theta * (\theta_1 - \theta_2)^2 + k_c * (c_1 - c_2)^2 + k_h * (h_1 - h_2)^2,$$

where ks are the weighting factors. A sequence alignment score is computed using a Dynamic Time Warping (DTW) procedure. The use of the total cost of Dynamic Time Warping as a similarity measure is helpful to group together strokes that are related to their root character by partial match. Dynamic Time Warping is a dynamic programming based procedure to align two sequences of signals. This can also provide a similarity measure.

Let the strokes (say their curvatures) are represented as a sequence of vectors F = F_1,F_2,...,F_M and G = G_1,G_2,...,G_N. The DTW-cost between these two sequences is $D_s\,(M, N)$, which is calculated using dynamic programming is given by:

$$D_s(i,j) = min \begin{cases} D_s(i-1, j-1) \\ D_s(i, j-1) \\ D_s(i-1, j) \end{cases} + \quad d(i,j)$$

where, $d(i,j)$ is the cost in aligning the ith element of F with jth element of G and is computed using squared Euclidean distance.

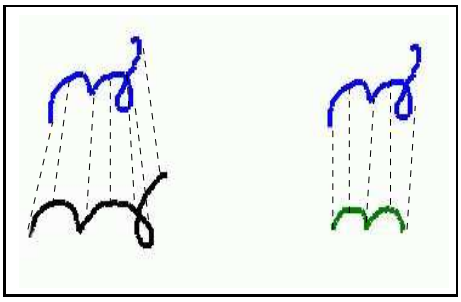

Figure 4. Illustrating the DTW-based matching of two strokes of Telugu character "e".

Using the given three values $D_s(i,j-1)$, $D_s(i-1,j)$ and $D_s(i-1,j-1)$ in the calculation of $D_s(i,j)$ realizes a local continuity constraint, which ensures no samples left out in time warping. As shown in Figure 5, we have also imposed global constraint using Sakoe - Chiba band [4] so as to ensure the maximum steepness or fatness of the DTW path. Score for matching the two sequences F and G is considered as $D_s(M,N)$, where M and N are the lengths of the two sequences.

3.4. Character and Word Level Annotation

We use the stroke matching module to come up with the best alignment of the strokes to the corresponding characters and hence words. As we pointed out earlier, we assume that each character in the text corresponds to one or more strokes in the input handwriting. Hence, annotation is the process of mapping a sequence of strokes from the input handwriting to the corresponding character. Once the characters are mapped, the segmentation and annotation of the words are straight forward. However, computing the best assignment of strokes to characters is not trivial as multiple strokes can form a character.

We employ a modified version of the elastic matching or dynamic time warping (DTW) algorithm to solve this problem. With the assumption that multiple strokes might map to a single character, we formulate the problem as follows:

Let $S = \langle s_1, s_2, \cdots, s_n \rangle$ be the stroke sequence in a word and let $C = \langle c_1, c_2, \cdots, c_m \rangle$ be the corresponding handwritten characters synthesized from the transcript. The problem is to find the best alignment between S and C, where $n > m$ and the cost of the best alignment is computed as:

$$D_w(c_i, s_j) = min \begin{cases} D_w(c_i, s_{j-1}) + Penalty(s_j) \\ D_w(c_{i-1}, s_j) + Penalty(c_i) \\ D_w(c_{i-1}, s_{j-1}) + D_s(c_i, s_{j,j}) \\ D_w(c_{i-1}, s_{j-2}) + D_s(c_i, s_{j-1,j}) \\ D_w(c_{i-1}, s_{j-3}) + D_s(c_i, s_{j-2,j}) \\ D_w(c_{i-1}, s_{j-4}) + D_s(c_i, s_{j-3,j}) \end{cases}$$

where, $D_s(c_i, s_{j,k})$ is the matching score obtained from the stroke alignment routine in aligning the ith character of C with j^{th} through k^{th} strokes of S. The feature vectors of the j^{th} through k^{th} strokes are concatenated to compute this matching score. The score for matching S and C is considered as $D_w(M,N)$, where M and N are the lengths of the stroke sequences in S and C respectively.

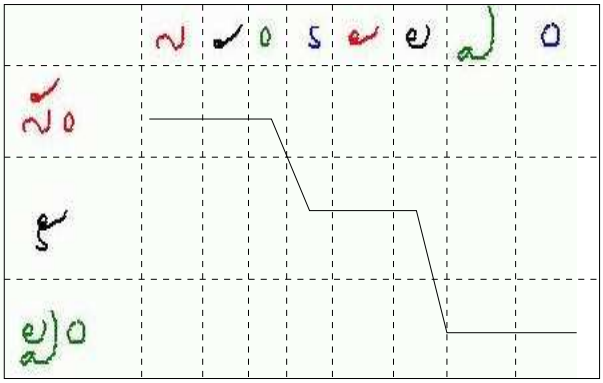

Figure 5. Figure (Matrix) Illustrating the word DTW. Horizontal continuous line segments show the strokes in columns grouped together to match with a character in a row. Continuous line segments across rows show the place of character boundaries.

Detecting Word Boundaries: It may be noted that the above algorithm assumes that the word boundaries are known for the input handwriting sequence. However, this is not the case as we described in the introduction. Hence we modify the above DTW algorithm to account for word boundaries. To compute the ending of a given word, w_i, in the text sequence, we augment the characters of w_i with one or more characters from the following word, w_{i+1}, at the end of the characters of w_i. Once the matching is performed, we identify the minimal distance of matching the augmented sequence with any sequence of strokes and then remove the strokes that mapped to the augmented characters. This method was found to be robust enough to detect the word endings in the experiments that we performed.

Updating the Handwriting Model: After the annotation is performed at the character level, we update our handwriting model by replacing the stoke models for the writer with samples from the current document, where the matching scores are high. Note that the replacement can be performed only for those characters that are present in the document. However, this is sufficient as the updated model is only used for the document under consideration. Once the handwriting model is updated, we use the updated model to carry out the annotation process. We repeat the cycle until there are no more changes to the result of annotation or a threshold is reached for the number of cycles.

Challenges in Online Annotation: Another interesting aspect to annotation on online data is the additional variability that the temporal information in the data introduces. For example, overwritings or corrections that are common in handwriting gets recorded as additional strokes in handwriting as opposed to modifications of the existing data. This introduces a challenge to our matching algorithm as the synthesized data does not contain the corresponding correction strokes. Our system can handle this to a large extent, since the erroneous strokes are often discarded by the matching procedure.

4. Storage and Reuse of Annotation

The format in which the annotation is preserved is important both for the use of the results for later algorithms as well as for the updation process described above. XML is the most widely accepted standard for data storage. A XML based annotation framework is advantageous both from the data representation view point and also from data storage view point. This allows users to add specific information to the annotation file to suit the needs of application at hand. Figure 6 shows a portion of the annotation, generated by our system that contains Devanagiri script written in both Unicode and ITRANS encoding schemes.

```
<labelTypes>
<labelType encoding="unicode">truth</labelType>
</labelTypes>
</labelSrc>
</labelSrcDefs>
<annotationDefs id="ID5">
<annotationScheme id="ID6">
<annotationLevel rank="1" name="WORD">description of WORD definition
 </annotationLevel>
<annotationLevel rank="2" name="CHARACTER">description of
CHARACTER definition
</annotationLevel>
</annotationScheme>
</annotationDefs>
</datasetDefs>
<hwData id="0" annotationSchemeRef="/descendant::
annotationScheme[attribute::id=ID6]">
<H1 id="ID7" writerRef="/descendant::writer[attribute::id=ID2]">
<label id="ID8" labelSrcRef="/descendant::labelSrc[attribute::id=ID4]"
labelType="truth" timestamp="2006-04-13T11:03:52.0Z">
<alternate rank="1" score="100">abhyass</alternate>
</label>
<H2 id="ID9" writerRef="/descendant::writer[attribute::id=ID2]">
<label id="ID10" labelSrcRef="/descendant::labelSrc[attribute::id=ID4]"
labelType="truth" timestamp="2006-04-13T11:03:58.0Z">
<alternate rank="1" score="100">a</alternate>
</label>
```

Figure 6. A sample UPX file used to store the annotation information for an Indian Language handwriting.

The UPX representation [7] makes use of an underlying XML representation of the raw handwriting data called the digital Ink Markup Language (InkML). The *labelType* tag in UPX contains the *encoding* attribute where the user can choose to have various encodings such as the Unicode, ITRANS or any other encoding formats. The annotated text is stored in the corresponding encoding within the tags *alternate* that is within the *label* tag. The *skillScript* tag contains the attribute *script* that can be used to store the script information of the writer. This annotated data can also serve as the meta information and can be further used in handwriting retrieval applications.

5. Experimental Results and Discussions

The experimental data for the annotation experiments were collected using a CrossPad. The data collected included 15 pages of handwritten data in Telugu script and one page of Hindi and Tamil scripts. The data was transcribed using ITRANS encoding to form a parallel text corpus. User models for synthesis were learned from part of the input data and the remaining were chosen for experiments on synthesis.

The text input is first converted into handwritten data using the synthesis module with the model of the user under consideration. The synthesized handwriting contains the annotation information at the character level. The word is then aligned with the original handwritten input and the annotations are propagated from the synthesized word to the original input word. In the process, word boundaries are also identified (see Figure 7).

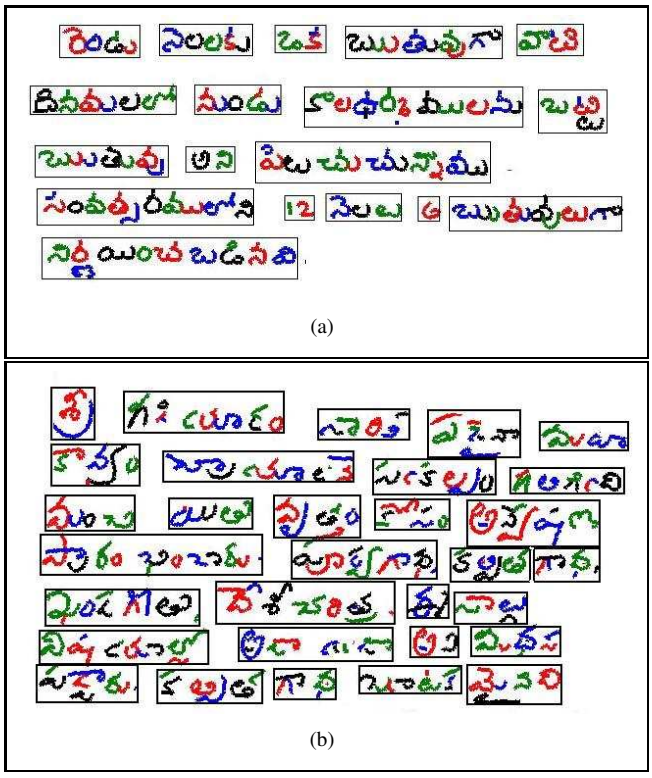

Figure 7. Word boundaries identified during the matching process. (a) Word boundaries in a constrained handwriting script. (b) Word boundaries identified in an unconstrained handwriting script.

In addition to the word boundaries, the annotation is propagated at the character level from the synthesized to to the original data. Figure 8 shows examples of Telugu words that were correctly annotated using our method.

The algorithm also allows for correction of errors that occur during the writing or transcription process. For example if the transcription was corrupted by a spurious character that changes the word form in the synthesized handwriting, the matching process often assigns no match to the spurious character, effectively removing the tran-

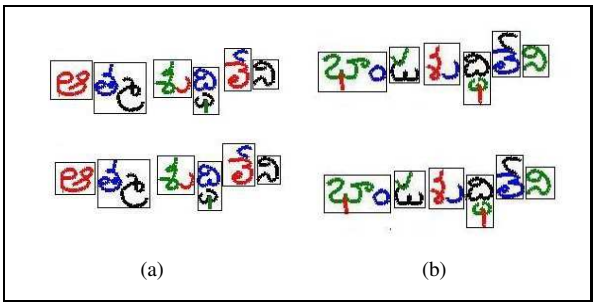

Figure 8. Annotation at character level achieved through matching. Illustrating the character boundaries identified by matching.

scription error from getting into the annotation. Figure 9 shows an example of an incorrectly transcribed word that was corrected during the matching process. The matching process can make errors in case of similar looking strokes that are to be matched. Figure 10 shows an example of a word that is incorrectly matched.

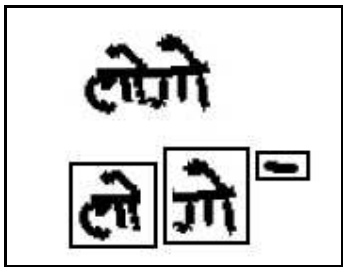

Figure 9. Correction of transcription: The synthesized word in Hindi at the bottom contains an extra stroke, which is discarded by the matching.

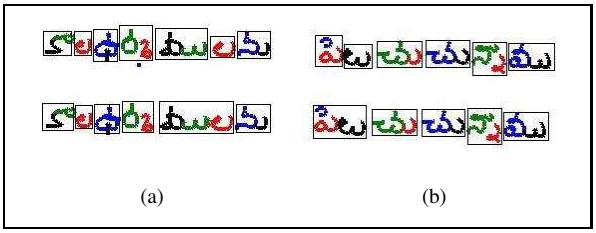

Figure 10. Errors in Matching: a) and b) shows examples of two words, where strokes in the synthesized word (top) were not matched with any in the original handwriting (bottom).

To compute a quantitative measure of accuracy of the proposed annotation scheme, we manually annotated our training corpus using the annotation toolkit [1]. The propagated annotations from the synthesis module was then compared with the manual annotation to find out the error rate in transcription. The error rate at the word level was 26.5%, tested over 425 words. However, the number could be misleading due to fact that word-level errors often arise due to an error in a single stroke being mislabelled in the data. The character-level annotation gives a better picture of the accuracy. The accuracy of character level annotation was 96.4% when tested on a set of over 3500 characters. In other words, most of the word errors were essentially single character errors.

6. Conclusions and Future Work

In this paper, we propose a model-based framework for annotation of non-cursive online handwritten data when a parallel text corpus is present for the data. The approach employs a handwriting synthesis scheme that generates the handwritten equivalent of the transcription. An elastic matching technique is used to propagate the annotation from the synthesized words to the original handwritten words. The annotation can be improved further by using the current annotation (partially correct) to refine the handwriting model of the writer under consideration, and then repeating the annotation process. Currently we are extending this work to incorporate partially cursive scripts under the same framework. Work is also being done to extend the labelling to stroke level. This would enable one to handle various stroke orders within the handwriting of a person. Currently it is assumed that a person has a consistent stroke order. This work is also being extended for printed character annotation.

References

[1] A. Bhaskarbhatla, S. Madhavanath, M. Pavan Kumar, A. Balasubramanian, and C. V. Jawahar, "Representation and annotation of online handwritten data," in *Proceedings of the International Workshop on Frontiers in Handwriting Recognition*, pp. 136–141, 2004.

[2] M. Zimmermann and H. Bunke, "Automatic segmentation of the IAM off-line database for handwritten english text," in *Proceedings of the 16th International Conference on Pattern Recognition*, pp. 35–39, 2000.

[3] C. Tomai, B. Zhang, and V. Govindaraju, "Transcript mapping for historic handwritten document images," in *Proceedings of the 8th International Workshop on Frontiers in Handwriting Recognition*, pp. 413–418, 2002.

[4] E.M. Kornfield, R. Manmatha, and J. Allan, "Text alignment with handwritten documents," in *Proceedings of the International Workshop on Document Image Analysis for Libraries*, pp. 195–209, 2004.

[5] J. Rothfeder, T.M. Rath, and R. Manmatha, "Aligning transcripts to automatically segmented handwritten manuscripts," in *Proceedings of the Seventh International Workshop on Document Analysis Systems*, pp. 84–95, 2006.

[6] Dave Elliman and Nasser Sherkat, "A truthing tool for generating a database of cursive words," in *Sixth International Conference on Document Analysis and Recognition*, pp. 1255–1262, 2001.

[7] Mudit Agrawal, Kalika Bali, Sriganesh Madhvanath, and Louis Vuurpijl, "UPX: a new XML representation for annotated datasets of online handwriting data," in *Proceedings of International Conference on Document Analysis and Recognition*, pp. 1161–1165, 2005.

[8] Guyon.I, Schomaker.L, Plamondon.R, Liberman.M, and Janet.S, "Unipen project of on-line data exchange and recognizer benchmarks," in *Proceedings of International Conference on Pattern Recognition*, pp. 29–33, 1994.

[9] C.V. Jawahar and A. Balasubramanian, "Synthesis of online handwritten data for Indian languages," in *Proceeding of the Tenth International Workshop on Frontiers in Handwriting Recognition*, 2006.