2011 International Conference on Document Analysis and Recognition

Baseline Dependent Percentile Features for Offline Arabic Handwriting Recognition

Pradeep Natarajan, David Belanger, Rohit Prasad, Matin Kamali, Krishna Subramanian, Prem Natarajan
Raytheon BBN Technologies
10 Moulton Street, Cambridge, MA 02138, USA
{*pradeepn,dbelange,rprasad,mkamali,ksubrama,pnataraj*}*@bbn.com*

***Abstract*—Handwritten text in Arabic and other languages exhibit significant variations in the slant and baseline of characters across words and also within a single word. Since the concept of baseline does not have a precise mathematical definition, existing approaches use heuristic methods to first identify a set of baseline relevant pixels and then fit lines/curves through them. However, for statistical features like percentiles that we use in our system, we only need an approximate curve that is close to the baseline to normalize the features. Hence we propose a two stage approach to estimate the approximate baseline. First we segment the text line into a set of components, and then estimate the baseline in each component using two methods - max projection and smoothed centroid line. We incorpate the computed baseline into percentile feature computation in the BBN Byblos OCR system for an Arabic offline handwriting recognition task. Our new features, result in a 1% absolute gain and 3.1% relative gain in the word error rate on a large test set with 15K handwritten Arabic words, which is statistically significant with p-value$<$0.001 using the matched pair comparison test. Further, our results show that computing fine-grained baselines from small line segments is significantly better than estimating a single baseline over the entire text line.**

***Keywords*-Feature Extraction, Baselin-dependent percentile, handwriting recognition**

I. Introduction

In recent years, advanced digital scanning technologies and cheap memory have become ubiquitous. This has resulted in the availability of large amounts of digitally scanned documents in many languages from different sources. Automated analysis and understanding of the information in these documents can have a wide range of applications. Despite significant progress in optical character recognition (OCR) and handwriting recognition (OHR) technologies, several challenges remain.

A key difficulty is the variation in baseline of handwritten text not only between words but even within a word. As a result, the shape of handwritten glyphs vary not only across different writers, but also across different instances written by the same writer. Thus, accurate estimation of the baseline is crucial for the performance of subsequent feature extraction and recognition. While the concept of baseline is intuitive to the human reader, it lacks a precise mathematical definition. The simplest approaches for baseline estimation are based on analyzing the horizontal projection histogram. However this is unsuitable for handwriting due to the inherent slant and skew in the text.

To address this, several methods have been proposed for baseline estimation [1][2][3][4]. These typically work by first identifying a set of pixels in the text that are near the location of the baseline, and then fitting a line/curve through them. In [1], baseline is estimated by first extracting a polygonally approximated skeleton of the word, identifying baseline relevant points in the skeleton and then fitting the baseline based on regression analysis. In [2], the minima points on the contour of the word is used to first fit an approximate baseline, which is then refined. In [3], baseline relevant points are identified based on template matching, and then a cubic polynomial is fit to get a baseline estimate. These approaches show encouraging results, but have had limited application in large scale Arabic handwriting recognition, with complications such as cursive writing and the presence of dots, strokes and other diacritic marks. Further, existing methods have been primarily tested on databases such IFN/ENIT[5] with pre-segmented words. However, word and character segmentation is in itself challenging.

Traditionally, an unsegmented line of text is first segmented based on distinctive features [6] and then the characters in each segment are recognized. However in recent years, hidden Markov model (HMM) based methods which automatically segment characters during recognition [7] have become popular due to improved performance. These approaches rely to a large extent on statistical features like percentiles [8], which are not normalized for baseline variations. We address this limitation using a novel feature, called baseline-dependent percentiles.

We describe two approaches to approximate the notional baseline. Intuitively, the vertical centroid of written text closely tracks the baseline in Arabic and is parallel to it in many languages such as English. Based on this, we introduce a new *centroid-based percentile (CPER)* feature that corrects for baseline variations, by normalizing percentile features with respect to the smoothed centroid track. We compare this with baselines estimated using max projection profiles (MPER) and the plain percentile (PER) features. Both CPER and MPER features show significant improvement over the percentile features for HMM based recognition on a large vocabulary, free-form Arabic handwriting corpus [9]. CPER which models the slant in text produces the biggest gain.

In the rest of the paper, we will first describe the BBN

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.74

offline handwriting recognition (OHR) system in section 2, then describe our baseline estimation techniques in section 3, then present our novel baseline dependent percentile features in section 4, give an overview of the corpus we used for testing and our experimental results in section 5, and conclude in section 6.

II. Overview of the BBN Byblos OHR System

The BBN Byblos OHR system we use is based on the work presented in [7]. This system models handwritten text as the output of HMM-based character models and has three modules: feature extraction, training and recognition.

Feature Extraction: This is the first step in both training and recognition. We convert 2-dimensional images to a 1-d feature sequence by first locating the top and bottom boundaries of the text lines, and then computing feature vectors in each line from a sequence of thin, overlapping vertical windows called *frames*. In the baseline system, we extract the following script-independent features from each frame: Percentiles of intensity values [8], Angle, Correlation, Energy (PACE) and GSC (Gradient, Structure and Concavity)[10] features. This set of features is called PACE+GSC and is described in detail in [8]. In our current system, we replaced the percentile features with features that correct for baseline variations.

Training: We model each character using a 14-state, left-to-right HMM, whose states model the output probability distributions over the features as a Gaussian mixture. We use context-dependent HMMs [11] to capture the fact that cursive characters' appearances depend on their neighbors. In total, we trained 339K Gaussians for 176 unique characters, which included Arabic characters, numerals, punctuations and English characters)

Recognition: The BBN Byblos recognition engine performs a two-pass search using glyph HMMs and a language model. We use a trigram language model trained on 90 million words of Arabic newswire data, with a 92,000 word lexicon and a test set out-of-vocabulary rate of 4.2%. We use a fast match beam search for the forward pass, using the HMMs and an approximate bi-gram language model. This outputs the most likely word-ends per frame. During the backward pass, we restrict the search space using the set of choices from the forward pass and an approximate trigram language model to produce an N-best list of hypotheses.

III. Baseline Estimation

The features in the system described so far are computed without taking into account significant baseline variations that are common in handwritten text. Hence, the feature computation is sensitive to style variations across different authors and can introduce significant errors. Further, baseline variations are seen not only across writers, but also among words and characters written by the same writer in a single line. However, computing the baseline to correct for such variations is hard, since character segmentations are not available during feature computation. Also, the notion of baseline in handwritten text is imprecise and cannot be defined mathematically.

Existing techniques for baseline estimation [1][2][3][4] can be thought of in terms of two distinct stages - *identification* of baseline relevant pixels in text, and *curve-fitting* through the identified pixels. A key challenge here is that the first stage relies on heuristic techniques, which make the approaches sensitive to parameter settings and difficult to generalize for large datasets. We address this by first segmenting the line into a set of components and then estimating the baseline for each component. This allows us to correct for baseline variations within a line robustly. We will discuss our approach next.

A. Line Segmentation

This involves segmenting a line image into a set of components, where we can ignore the intra-component baseline variations. Given that we do not have character segmentations apriori, identifying such components is challenging. The simplest approach is to treat the entire line as a single component, and estimate a baseline over the line image. This would allow us to correct for line-level slant that is seen in many writers.

However, the baseline estimated from this simple approach is imprecise and does not account for inter-word and inter-character variations. To address this, we first detect *connected components* in the line of text and estimate a baseline for each component. This allows fine grained analysis of the baseline, but has one key limitation - the dots and other diacritic marks are typically in separate connected components and can confuse the baseline computation. To address this we eliminated any connected component c_1 if there existed another component c_2 such that:

$$c_1.l \geq c_2.l \wedge c_1.r \leq c_2.r \wedge c_1.pts \leq c_2.pts \quad (1)$$

where, $c.l$, $c.r$ are the locations of the left and right horizontal extremes and $c.pts$ is the number of points in connected component c. Thus, we in effect eliminate those components c_1 whose horizontal extent is completely contained in the horizontal extent of another component c_2. Intuitively, such components c_1 correspond to dots and diacritics.

However, it is possible that components corresponding to such marks can extend beyond the horizontal extent of the corresponding character's component. Further, components from different words/PAWs can also overlap horizontally and having multiple baselines at a given x location can complicate downstream feature extraction. We could address this by merging components (c_1,c_2) whose horizontal extents overlap or are nearby:

$$\frac{\min(c_1.r, c_2.r) - \max(c_1.l, c_2.l) + 1}{\max(width(c_1), width(c_2))} \geq O_{Th}$$

where $width(c)$=$c.r$-$c.l$+1 is the horizontal width of component c. Positive values for O_{Th} ensure merging of only those components that overlap horizontally, while negative values merge even nearby components. However, merging such components can result in coarser baseline estimates as the individual components in the larger component can have different baselines. Figure 1 illustrates this.

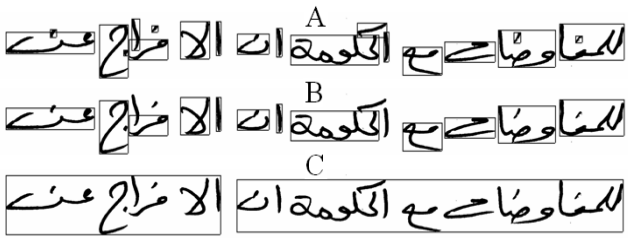

Figure 1. A: Extracted Connected Components B:Eliminate dot-like components using eq. 1 C: Merge neighboring components using eq. 2, with O_{Th}=-1

B. Baseline Computation

We considered two possible methods for estimating the notional baseline after segmenting the line into constituent components.

Max Projection: In this method, we first traverse each component c vertically and count the number of text pixels at each vertical location y:

$$profile_c(y) = \sum_{x=c.l}^{c.r} f_c(x,y) \quad \forall y \in [c.t, c.b] \tag{2}$$

where $f_c(x,y)$=1 if there is text at pixel (x,y) belonging to component c and 0 otherwise. $c.t$ and $c.b$ are the top and bottom y locations of the component c respectively. Given these *projection profiles*, the baseline for component c is given by the line:

$$y_{baseline}(c) = \max_y \{profile_c(y)\} \quad \forall x \in [c.l, c.r] \tag{3}$$

The max-projection approach is simple, but has been effective in several applications including line finding [12]. However, it produces a horizontal line to estimate the baseline and hence is not robust to slant in handwriting.

Smoothed Centroid Line Estimation: Here, we traverse each component c horizontally and compute the vertical centroid of the text pixels at each x location:

$$y^c_{centroid}(x) = \frac{\sum_y y f_c(x,y)}{\sum_y f_c(x,y)} \quad \forall x \in [c.l, c.r] \tag{4}$$

After computing the vertical centroids, we estimate the baseline to be the least squares fit over $(x, y^c_{centroid}(x))$:

$$arg\max_{m,c} \sum_{x=c.l}^{c.r} ||y^c_{centroid}(x) - (mx + c)||^2 \tag{5}$$

This allows us to compute the slant in each connected component. Figure 2 illustrates the baselines estimated by different approaches. Triangles, crosses, and boxes denote maxprojection, centroid-based, and human annotation, respectively.

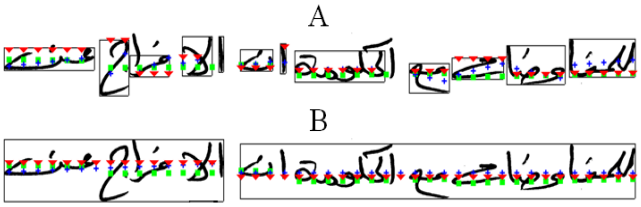

Figure 2. A: Baseline estimation with components from (A) dot elimination using (1), (B) merging with (2) $O_{Th} = -1$

C. Accuracy of Estimated Baseline

Since the baseline is an imprecisely defined concept, we evaluated the accuracy of our baseline estimation by comparing them to human annotations. We collected a set of 200 lines of handwritten Arabic text, and annotated them using Arabic speakers. The annotators were asked to take into account baseline variations across words and parts of Arabic words, but no other specific instructions were given.

Table I compares max projection and centroid line based baseline estimation, when the text lines were segmented by merging contained connected components (1), neighboring components (2) and by considering the entire line as a single component. The entries in the table correspond to the mean and variance of the relative pixel distances of the computed baseline w.r.t the annotations, on images with character height ≈250 pixels.

Method	Dot filtered (equation 1)	Merged Components (equation 2) O_{Th}=			Line
		-0.2	-0.5	-1.0	
Max Projection	12±11	13±13	18±22	20±23	20±24
Centroid Line	8±4	9±6	12±16	12±17	12±18

Table I
BASELINE ESTIMATION ERROR(%). MORE NEGATIVE VALUES FOR O_{Th} IMPLIES MORE AGGRESSIVE MERGING OF CONNECTED COMPONENTS. IN PRACTICE $O_{Th}\leq$-2.0 MERGES ALL COMPONENTS IN A LINE

Our study suggests that all the baseline estimation techniques we considered produce baselines which are offset by only ≈10-20% on average, from the annotated ground truth. Further, to correct feature computation we are mainly interested obtaining curves that are at a constant offset from the ground truth. Hence the variance is a more important measure. On both measures, the centroid line approach performs better. Also, the dot-filtered component extraction in the first column produces the best results. The same combination also produces the best performance in our recognition experiments in section 5.

IV. Baseline Dependent Percentile Features

After estimating the baseline, we incorporate into the computation of percentile features. These features are computed by integrating the number of text pixels from top to bottom, from a sequence of overlapping windows called *frames*, for each line of text [8]. Since there is significant variation in the slant and also whitespace above and below the line of text, we must first tighten the upper and lower boundaries of each frame. We do this by expanding the width of the frame w_f on both the left and right sides so that $w_t.width$=5$w_f.width$, and the upper and lower boundaries of the frame is defined by the bounding box of the text pixels in w_t. This is explained in detail in [9].

Once we tighten the frame, we compute the baseline-dependent percentile features as follows:

1. At each frame, we find the largest component that overlaps with the frame.
2. Next, we find the point on the computed baseline line, at the center of the frame.
3. Then, we divide the frame into two parts, those above the center point and those below.
4. Finally, we compute percentiles in each of the two parts, starting from the center line and concatenate the feature vectors.

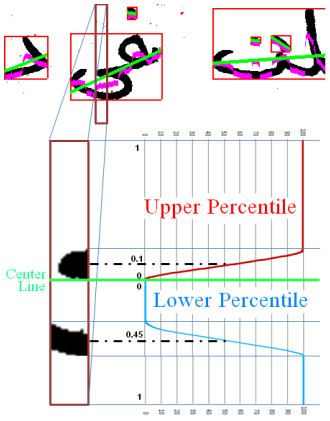

Figure 3. Computing baseline-dependent percentile features

This is illustrated in Figure 3. Thus, we are in effect computing two sets of percentiles for each frame: for text pixels above and below the estimated baseline. The percentiles integrate "Blackness", from the baseline to the top or bottom of the tightened frame. Thus, in figure 3, 50% of the blackness is in the first 10% of the upper window, while 50% of the blackness is in the first 45% of the lower window.

V. Experimental Results

Corpus Description: We use handwritten data collected by the Linguistic Data Consortium, in our experiments to test our features. The data consists of scanned images of handwritten Arabic text, from different scribes, with varied writing conditions. The ground truth annotations included word bounding boxes and the corresponding tokenized transcriptions. The details of our training, development (dev), and test sets are shown in Table II. There is no overlap of documents between these sets. Also, our development and test sets contained an equal number of documents from scribes previously seen and unseen during training. Our corpus, to the best of our knowledge, is largest collection of free-flowing Arabic handwritten documents with annotations. The data exhibits several characteristics that make text recognition hard, such as overlapping line/word boundaries, non-linear baseline within lines/words, slant, scratches and poor legibility.

Set	#Images	#Scribes	#Words
Train	9714	71	1389K
Dev	150	50	15K
Test	150	42	15K

Table II
Description of Train, Dev and Test Sets

Performance Comparison: For our experiments, we first trained a baseline PACE+GSC system using the features and setup we described in section 2. Then we replaced the percentile features in this system with our baseline-dependent percentile features computed using max projection and smoothed centroid line on connected components merged as in equation (1) - we call these the MPACE+GSC and CPACE+GSC systems respectively and repeated the training experiment. Then we tested PACE+GSC, MPACE+GSC and CPACE+GSC systems on the test data. Table III summarizes the word error rate of the systems. Our results show that our features which correct for baseline and slant variations produce significant improvements in WER over the PACE+GSC system.

Technique	Overall	Writers in Training	Writers not in Training
PACE+GSC	32.6	31.1	34.1
MPACE+GSC	32.1	30.8	33.5
CPACE+GSC	**31.6**	**30.1**	**33.2**

Table III
Summary of Results

Comparison of Component Extraction Methods: We also compared the relative performance of different component extraction methods, including those which merge fully contained connected components (1), and those which treat the entire text line as a single component. Table IV shows the performance of these approaches. The relative performance closely tracks the accuracy of baseline estimation in Table I, with the approach that estimates the baseline at the line

No baseline (PACE+GSC)	Dot filtered (equation 1)	Line
32.6	**31.6**	32.4

Table IV
COMPARISON OF DIFFERENT COMPONENT EXTRACTION METHODS FOR CPACE+GSC

level producing a minimal gain over PACE+GSC.

Significance Test and Robustness of Gains: Using the matched pair comparison test [13], we found the gains of CPACE+GSC over PACE+GSCE to be significant with a p-value$<$0.001, which is well below the traditional threshold of 0.05. We repeated our experiments over a range of windows for tightening each frame's upper and lower boundaries in figure 3, according to the approach described in [9]. Increasing the window for frame tightening improved the performance of both systems, but the CPER features consistently produced a 0.7%-1% gain in all cases. We repeated our experiments using the *page style adaptation* technique presented in [14] and got similar improvements in WER.

Figure 4, presents examples of handwritten lines where the CPACE system performed at least 60% better in WER than the PACE system. These documents typically contain significant skew and vertical drift of the baseline.

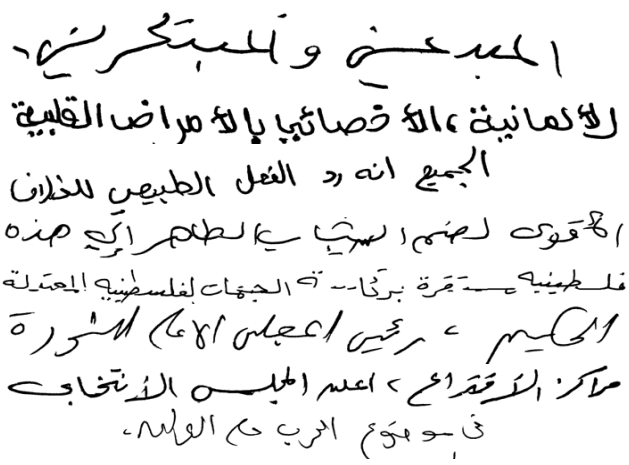

Figure 4. Examples where CPACE improvement is $>$ 60% WER

VI. CONCLUSION

We have presented a feature that helps correct for slant and baseline variations in handwritten text, which helps in eliminating the effects of noise and local stroke curvature. The estimated baselines approximate the track of the writer's baseline, within ≈8% of the image height. Normalizing our features with respect to this track consistently produces a statistically significant ≈1% absolute improvement in recognition accuracy.

Acknowledgement: This paper is based upon work supported by the DARPA MADCAT Program. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the US Government.

REFERENCES

[1] M. Pechwitz and V. Märgner, "Baseline estimation for arabic handwritten words," in *IWFHR*, 2002, pp. 479–.

[2] F. Farooq, V. Govindaraju, and M. Perrone, "Pre-processing methods for handwritten arabic documents," in *ICDAR*, 2005, pp. 267–271.

[3] M. Ziaratban and K. Faez, "A novel two-stage algorithm for baseline estimation and correction in farsi and arabic handwritten text line," in *ICPR*, 2008, pp. 1–5.

[4] H. Boukerma and N. Farah, "A novel arabic baseline estimation algorithm based on sub-words treatment," *ICFHR*, vol. 0, pp. 335–338, 2010.

[5] M. Pechwitz, S. S. Maddouri, V. Margner, N. Ellouze, and H. Amiri, "Ifn/enit-database of handwritten arabic words," in *7th Colloque International Francophone sur l'Ecrit et le Document, Hammamet, Tunis*, 2002.

[6] G. Kim and V. Govindaraju, "A lexicon driven approach to handwritten word recognition for real-time applications," vol. 19, no. 4, pp. 366–379, 1997.

[7] P. Natarajan, S. Saleem, R. Prasad, E. MacRostie, and K. Subramanian, "Multilingual offline handwriting recognition using hidden markov models: A script-independent approach," *Springer Book Chapter on Arabic and Chinese Handwriting Recognition*, vol. 4768, pp. 231–250, 2008.

[8] P. Natarajan, Z. Lu, R. M. Schwartz, I. Bazzi, and J. Makhoul, "Multilingual machine printed ocr," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 15, no. 1, pp. 43–63, 2001.

[9] S. Saleem, H. Cao, K. Subramanian, M. Kamali, R. Prasad, and P. Natarajan, "Improvements in bbn's hmm-based offline arabic handwriting recognition system," in *ICDAR*, 2009, pp. 773–777.

[10] S. Tulyakov and V. Govindaraju, "Probabilistic model for segmentation based word recognition with lexicon," in *ICDAR*, 2001, pp. 164–167.

[11] R. Prasad, S. Saleem, M. Kamali, R. Meermeier, and P. Natarajan, "Improvements in hidden markov model based arabic ocr," in *ICPR*, 2008.

[12] S. Calabretto and A. Bozzi, "The philological workstation bambi (better access to manuscripts and browsing of images)," *Journal of Digital Information (JoDI)*, vol. 1, no. 3, 1998.

[13] D. Pallet, W. Fisher, and J. Fiscus, "Tools for the analysis of benchmark speech recognition tests," *ICASSP*, vol. 1, pp. 97–100, 1990.

[14] H. Cao, R. Prasad, S. Saleem, and P. Natarajan, "Unsupervised hmm adaptation using page style clustering," in *ICDAR*, 2009, pp. 1091–1095.