Portable Language-Independent Adaptive Translation from OCR

Quarterly R&D Status Report No. 8

October 15, 2009

Contractor:	**BBN Technologies** 10 Moulton Street, Cambridge, MA 02138
Principal Investigator:	Prem Natarajan Tel: 617-873-5472 Fax: 617-873-2473 Email: pnataraj@bbn.com
Reporting Period:	1 July 2009 – 30 September 2009

This material is based upon work supported by the Defense Advanced Research Projects Agency
DARPA/IPTO
Portable Language-Independent Adaptive Translation from OCR
MADCAT Program
ARPA Order No: X 103
Program Code: 7M30
Issues by DARPA/CMO under Contract #HR001-08-C-0004

Executive Summary

This is the eighth R&D quarterly progress report (QPR) of the BBN-led team under DARPA's MADCAT program. This report is organized by technical task area.

1.1. Pre-Processing and Image Enhancement [BBN, Polar Rain, SUNY, UMD]

Improved Rule Line Cleaning and Restoration [Polar Rain]: This quarter, we re-designed the Shape-DNA based rule line cleaning algorithm to minimize the degradation of the shape of text characters. Recall that in the Shape-DNA based cleaning approach, the projection onto the Shape-DNA space produces a rule line distance image that is used to clean the rule lines. However, this cleaning process can and does remove portions of legitimate text characters that resemble rule lines. Therefore, instead of using the rule line distance images for directly cleaning rule lines, we now use this image to *model* the rule lines present in the document. Specifically, by applying Hough transform to the rule line distance image, we compute a set of model parameters. In addition, we estimate the average thickness of the rule lines using the original input image. Finally, we use both the rule line model parameters and the rule line thickness information with a sliding window to clean the rule lines. Figure 2 shows an example where the performance of the new rule line cleaning algorithm is compared with the performance of the previous version of the shape-DNA cleaning.

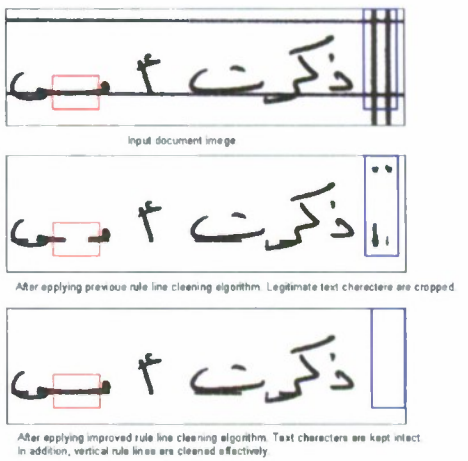

Figure 1: Improved Shape-DNA cleaning.

This reporting period, we also improved the restoration algorithm for removing the artifacts introduced by rule line cleaning. Similar to rule line cleaning algorithm, Shape-DNA based restoration algorithm also includes an off-line training process, where text characters' shapes are learned off-line by training about 100 handwritten text images (with no rule lines) and a Shape-DNA database is computed from the shape patterns. These shape patterns are then used in the restoration of text characters in the rule line cleaned images by projecting the shape blocks from the input image onto the database and by searching for the closest shape pattern in the database. Unlike our previous version, where shape-DNA restoration was applied to entire image, we now use the estimated rule line model parameters to constrain the restoration into the local proximity of detected rule lines.

1.2. Page Segmentation [BBN, Polar Rain, UMD, SUNY]

Line Segmentation using Baselines [Polar Rain]: We improved text line detection for handwritten documents so that text characters are assigned to the correct baseline and to improve the accuracy of detection of small diacritics. We first detect baselines to which text characters are anchored, and then assign each text character to the appropriate baseline. During the assignment process, we limit the horizontal space between consecutive text characters that are assigned to a particular baseline so that when two adjacent baselines overlap vertically text characters are not assigned to the wrong baseline. Figure 6 shows an example where the performance of the revised line detection algorithm is compared with the previous version.

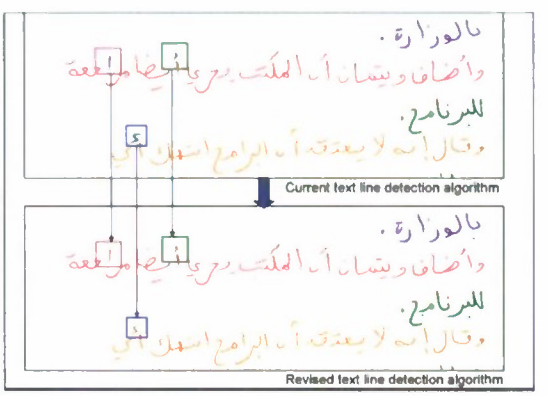

Figure 2: Improved line segmentation using baselines.

Line Segmentation using Filter-Banks and Graphs [UMD]: This reporting period, we designed and implemented a line segmentation algorithm which is more robust to characteristics of real-world data such as the Anfal corpus. Our algorithm is based on a combination of filter-banks and graph segmentation. Specifically, the first stage of the algorithm applies a bank of anisotropic Gaussian filters of different orientations and scales. The orientation and scale parameters of the filter bank are estimated from the underlying image directly, after initial preprocessing and noise removal steps. The second stage models the document as an undirected weighted graph,

where each connected component is represented by a node in the graph. To segment the graph, we decided to re-use our previously implemented Affinity Propagation (AP) method. The advantage of the using AP is that the number of sub-graphs need not be a priori specified.

Another novel feature of our algorithm is robust estimation of the baselines of the text lines. In order to estimate the baselines, foreground pixels of all connected components that belong to the segmented sub-graph are used to estimate line parameters. We use a maximum likelihood (ML) variant of the popular RANdom SAmple Consensus (RANSAC) for estimating the baselines. Also at this stage, diacritics that were removed during the preprocessing stage are assigned to the text lines based on two criteria: (1) vertical distance between the diacritic centroid and the baselines and (2) inclusion of the diacritic into the convex hull of the segmented text line.

1.3. Text Recognition [BBN, Argon, Columbia, SUNY]

Novel Features for Text Recognition [BBN, SUNY]: In this reporting period, BBN and SUNY explored multiple novel features for improving text recognition.

Centroid-based Percentile Features [BBN]: The percentile features computed in our handwriting recognition system assume that there is a fixed, horizontal baseline for the characters on a text line. However, real-world handwritten documents exhibit variations in skew, slant, and baselines within a text line and oftentimes with a word itself. Therefore the features used for recognition should be robust to such variations in the baseline. Since defining and estimating a *precise, consistent* baseline for a text line or constituent words that normalizes for first order variations in shape across the different instances of a particular word or sub-word or character is difficult to do if not impossible, we introduce the notion of *centroid* which is computed as follows. First, we use connected component analysis to separate contiguous word segments. We also remove components which are smaller in size than a specified threshold. Next, for each remaining connected component, we scan the component horizontally and compute the centroid of black pixels at each vertical strip / analysis window. Finally, we compute percentile features both above and below the centroid for each analysis window.

To assess the utility of the above centroid-based percentiles, we trained two hidden Markov model (HMM) based systems. The first system was trained with standard percentile features (PACE) and Gradient-Concavity (GC) features, whereas the second one used centroid-percentiles (CPACE) and GC features. Initial recognition experiments on a training set of 20K images shows an absolute improvement of 0.5% in word error rate (WER) for using CPACE+GC features over our standard system that is trained with PACE+GC features.

Gabor Features [BBN]: 2-D Gabor filters are widely used for texture analysis of images due to their local band-pass and localization properties in both the spatial domain and the spatial frequency domain. In this reporting period, we explored Gabor-filter based features for text recognition. First, we apply 4 Gabor filters to the entire text line image, with each filter representing an orientation direction (horizontal, vertical, positive and negative 45° slope with respect to the horizontal axis). Next, from the real part of the filtered image, we compute the sum of positive values in each bin within an analysis frame divided by the area of the bin. Note that the sizes of analysis frames and bins within the frame are the same as those of the GC features, i.e. 12 bins per frame, and the width and the height of each bin equal $1/12^{th}$ of the "effective" line height as described in the previous quarterly report. The imaginary part of the filtered image is not used.

Early assessment of the Gabor features shows that they perform as well as the Gradient-Structure-Concavity (GSC) features for the 34 Parts-of-Arabic word (PAW) using support vector machines. The top-1 classification accuracy for using Gabor features was 82.7% compared to 81.6% for the GSC features. Preliminary recognition experiments on the MADCAT data using our HMM system trained with PACE+GSC+Gabor features shows a 0.5% improvement in WER over the standard PACE+GSC system.

Stroke Features [SUNY]: This reporting period, we explored novel features that represent medium to coarse level stroke shapes for stroke endings, sharp turnings, inner loops and small dots and 4 directional features for the stroke directions. Our implementation uses a stroke following approach based on tracing the contour structure so as to model the generation of the input text. An input word image is first converted into the contour code as consecutive arrays of boundary pixels. Tracing the contours counter-clock-wise, an angle attribute is calculated for each contour point using its neighboring contour pixels, from which the stroke turning is decided. A left turn indicates a stroke ending and a right turn indicates a sharp curve which is either an inner curve of a letter or a joint between two letters. The loops and isolated dots are also determined by the sizes of the connected components in terms of their inner contours and outer contours. Preliminary experiments on the 34-PAW data set

using stroke features show comparable performance to the GSC features. We are currently assessing the usefulness of combining these features with the PACE and GSC features.

Scribe Adaptation for Glyph Modeling [BBN]: This quarter, we explored scribe-adaptation for improving performance for both HMM and stochastic segment model (SSM) based glyph modeling.

Scribe-adapted HMMs [BBN]: In this approach, during training we estimate a scribe-adapted model for each scribe by adapting the "global" scribe-independent HMMs on pages written by the specific scribe. For adapting the scribe-independent model we use maximum a posteriori (MAP) technique. In addition, using stylistic features such as stroke width, slope, and contour features to train a k nearest neighbor (NN) classifier for scribe identification (kNN). During recognition each test page is first classified as being written by one of the scribes in training. Next, the scribe-adapted glyph HMMs for the top-choice scribe is used to recognize the page. Following the initial recognition, as in the standard system, we use the recognized transcriptions to perform maximum likelihood linear regression (MLLR) based *page-wise* adaptation.

We measured the efficacy of the above approach by performing training and recognition on the LDC data using line segmentation. A total of 37,608 pages from 259 unique scribes were used for training a global scribe-independent model and 259 scribe-adapted models. All models have an average of 2.5 million Gaussians. Also, the closed-set scribe ID performance on 259 scribes using kNN classifier is 59%. Next, we performed recognition experiments performed on a validation set consisting of 885 pages from 47 unique scribes, of which 23 scribes are not in training. As shown in Table 1, the WER for using the scribe-adapted models results in a 1% absolute improvement in WER over scribe-independent system. As one would expect, the improvement in WER with scribe-adapted HMMs is larger for scribes in training data. In the current system we perform a forced-choice scribe-assignment without any rejection mechanism (for example, a threshold on the scribe-similarity score). Therefore one might expect that using the scribe-independent models might perform better than using scribe-adapted models from the top-choice scribe (which is clearly the wrong scribe label for the page). However, analysis of performance on pages written by scribes not in training shows that there is no degradation in WER for using scribe-adapted models instead of scribe-independent models. We believe that this lack of degradation is due to two factors: the efficacy of the scribe-similarity measure and the large diversity of training scribes that provides an adequately fine exemplar-based quantization of styles.

Scribe-adapted Stochastic Segment Models, or SSM's, [BBN]: Recall that in Phase 1, we had designed a novel framework which we call stochastic segment modeling for integrating different types of features and classifiers. In the SSM framework the HMM is first used to generate 2-D character images (i.e. stochastic segments) for each n-best hypothesis. Subsequently, segment model classifiers, which in our current implementation are support vector machines (SVMs) trained with GSC features, are used to produce a segmental score. These segmental scores are used as additional knowledge source for rescoring the n-best list. This reporting period we performed similar experiments with SSM adaptation as ones reported above for adapting HMMs. For our initial experiments, instead of adapting a "global" SVM trained on stochastic segments from the entire dataset, we experimented with scribe-dependent support vector machines (SVMs) that are trained on stochastic segments only from a specific scribe. Instead of using the global SVM for n-best rescoring in the SSM framework, we now use the scribe-dependent SSMs for rescoring. Preliminary results from these experiments on a smaller training set (20K images instead of 37K pages) show an absolute improvement of 1% in WER (30.7% vs. 29.6%) over the un-adapted SSM framework.

System	Segment.	%WER
Phase 1 Eval	Word	31.0
Phase 2 scribe-ind.	Line	26.5
Phase 2 scribe-adapted	Line	25.5

Table 1: Performance of scribe-adapted HMMs and summary of improvements over Phase 1 system.

Probabilistic Graph Matching Improvements [Argon, BBN]: This reporting period, Argon delivered a Variable Basis Kernel (VBK) based graph comparator to BBN for integration and testing. During the integration process, Argon and BBN worked together on parallelizing the graph training across classes to reduce the training time. The parallelization of VBK training reduced the training time by a factor equivalent to the number of classes. Next, to assess the utility of probabilistic graph matching using the VBK character comparator, we performed experiments on stochastic segments generated from BBN's HMM engine. Specifically, we created a closed-set training and test sets of 50 classes from the stochastic segments. The training set comprises of 500 examples for each class for a total of 25K character images, whereas the test set comprises of 50 (held-out)

examples for each class for a total of 2.5K character images. The VBK engine from Argon resulted in a top-choice accuracy of 40.7%; in comparison, the performance of SVMs trained with GSC features on the same data resulted in an accuracy of 58.5%.

1.4.Integration with GALE MT [BBN]

This reporting period, we performed experiments with the latest AGILE MT engine. In addition, we started training MT models to support the MADCAT evaluation. Next quarter, we will use these engines to decode OCR output for system combination.

1.5.Metadata Extraction [BBN, BAE, Lehigh, SUNY, UMD]

Logo Detection and Recognition [BAE]: This quarter, we continued development of the logo detection algorithms, for both the constrained and unconstrained detection scenarios. The unconstrained problem consists of detecting logos and logo-like regions in document imagery, whereas the constrained detection problem consists of detecting the presence of any logo from a library of known logo classes in the document imagery. We extended our recognition approach for use in constrained logo detection, and generated detection performance results. For unconstrained logo detection, we developed an approach using morphological pre-processing and exploitation of pixel density and contour characteristics. Preliminary detection results were generated for performance evaluation and to gain insight into algorithm behavior.