

Versatile Search of Scanned Arabic Handwriting

Sargur N. Srihari, Gregory R. Ball and Harish Srinivasan

Center of Excellence for Document Analysis and Recognition (CEDAR)

University at Buffalo, State University of New York

Amherst, New York 14228

{srihari, grball, hs32}@cedar.buffalo.edu

Abstract

Searching scanned handwritten documents is a relatively unexplored frontier for documents in any language. In the general search literature retrieval methods are described as being either image-based or text-based with the corresponding algorithms being quite different. Versatile search is defined as a framework where the query can be either a textual string or an image snippet in any language and the retrieval method is a fusion of text- and image-retrieval methods. An end-to-end versatile system known as CEDARABIC is described for searching a repository of scanned handwritten Arabic documents; in addition to being a search engine it includes several tools for image processing such as line removal, line segmentation, creating ground-truth, etc. In the search process of CEDARABIC the query can be either in English or Arabic. A UNICODE and an image query are maintained throughout the search, with the results being combined by an artificial neural network. The combination results are better than each approach alone. The results can be further improved by refining the component pieces of the framework (text transcription and image search).

1 Introduction

While searching electronic text is now a ubiquitous operation, the searching of scanned printed documents such as books is just beginning to emerge. The searching of scanned handwritten and mixed documents is a virtually unexplored area.

Processing handwritten Arabic language documents is of much current interest. One unsolved problem is a reliable method, given some query, to search for a subset among the many such documents, similarly to searching printed documents. The problem is challenging because of the unique structural features of Arabic script and the relative infancy of the field of handwriting processing.

Content-based information retrieval (CBIR) is a broad topic in information retrieval and data mining

[1]. CBIR algorithms are quite different for the tasks of text retrieval and image retrieval. Correspondingly there are two approaches to searching scanned documents, stemming from the two different schools of thought. One approach is to use direct content based image retrieval (word spotting). Another is to convert the document to an electronic textual representation (ASCII for English and UNICODE for Arabic) and search it with text information retrieval methods used routinely with electronic documents. Both of these approaches can be successful under ideal circumstances, but such a situation is difficult to achieve with current technology of handwriting recognition. Image based searches do not always return correct results. Arabic handwriting recognition technology does not come close to allowing full transcriptions of unconstrained documents. However, by combining these two methods together, we achieve better performance than either on its own.

The paper describes a framework for versatile search of Arabic handwritten documents. By versatile search, we mean both versatility in the query and versatility in the search strategy—combining content based image retrieval and text-based information retrieval. Versatility in the query refers to the query being either in textual form or electronic form. Another characteristic of versatile search is that the query can be in multiple languages such as English and Arabic. In the versatile search process both the original scanned image and the (partial) transcription are maintained at all stages. Searches proceed in parallel on both document representations. Any query is also split into both an image and a UNICODE representation which act on the corresponding instance of the document. The results from both parallel searches are combined into a single ranking of candidate documents.

The rest of the paper is organized as follows. Section 2 describes previous work in scanned document retrieval. Section 3 describes the nature of queries for versatile search. Section 4 describes the overall

system architecture as well as the process of segmentation necessary for image indexing and retrieval. Section 5 describes the image-based word shape matching approach. Section 6 describes the text-based approach that is based on character recognition. Results in the form of precision-recall with a corpus of handwritten Arabic are described in Section 7. Conclusions and future directions are given in Section 8.

2 Related Work

Searching scanned printed documents in English has had significant success. Taghva et al showed [13] that information retrieval performance continues to be high even when OCR performance is not perfect. Russell et al [6] note this can, at least in part, be attributed to redundancy and the fact that while OCR performance may commit some errors, it performs very well for English. They go on to discuss the use of handwritten and typed queries. Arabic recognition technology, however, generally performs significantly poorer than its English counterpart.

A system for directly searching scanned handwritten English handwriting was discussed in [7]. This system, known as CEDAR-FOX, was developed for forensic document analysis applications [12],[8]. Searching scanned Arabic text within a system known as CEDARABIC was first reported in [9]. The CEDARABIC system was based on an end-to-end software system that was previously developed for English handwriting known as CEDAR-FOX. Both systems are designed to be interactive for use by a human document examiner and have many pre-processing operations such as line and word segmentation, rule-line removal, image enhancement, etc.

3 Queries and Searches

The query can take several forms: (i) a UNICODE string of Arabic text (for example, entered on an Arabic keyboard), specifying a word or words the user wants to appear in the handwritten document, (ii) an English word or words corresponding to an idea that should appear in the Arabic document, (iii) an image of an Arabic word or words; documents should be returned that also have a representation of this Arabic word.

Word spotting algorithms start with an image query; either a full word or component prototype characters. The document is searched directly, with only minor preprocessing steps such as noise removal, etc. We take two approaches for word spotting: word shape and character shape based. In the word shape based method, features are extracted from prototype word images. This prototype can either be provided with the query, or can be looked

up from a library of images based on a keyword. We then compare the features of a candidate word to the prototype words, choosing the best match. The character shape based method splits a candidate word into sequences of candidate component characters. Each sequence is matched to prototypes of the characters in the query word, and the sequence of candidate characters containing the overall maximum similarity to the prototype characters receives the highest score for that word, with the score acting as a confidence measure. The word shape based method performs well when many prototype images are present, but cannot be used if there are none available; at that point the character based method is the only available approach. In situations where both methods are applicable, their rankings are combined.

To partially transcribe documents we use several approaches of word recognition. A baseline, simple method is to perform a variation of character recognition and try to directly deduce a word. Arabic has an advantage over languages such as English because of the presence of subwords which are predictably distinct. In a second method, we compare the candidate characters against those suggested by a lexicon of words, choosing the candidate representation with the best score. Since larger lexicons generally result in poorer performance we limit the lexicon size when using such a method.

4 Framework

Figure 1 gives an overview of the versatile search framework. The key point is that both image and text queries are maintained against image and text versions of the document throughout the searching process, with their results in the end being combined with a neural network.

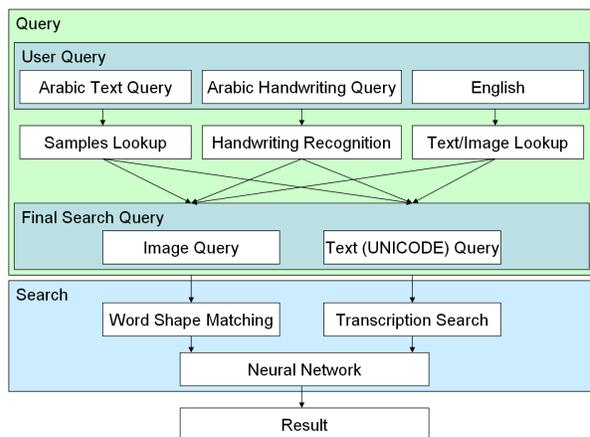


Figure 1: Versatile Search Framework

A critical common preprocessing step necessary

for both methods is segmenting a page into lines, and sometimes a line into words. A CEDARABIC representation of a segmented document is shown in Figure 2.



Figure 2: Segmented Document

4.1 Segmentation Algorithms

Automatic word segmentation, as presented in [10], is based on taking several features on either side of a potential segmentation point and using a neural network for deciding whether or not the segmentation is between two distinct words. Some of the differences between the tasks of segmenting Arabic script and segmenting Latin script are the presence of multiple dots above and below the main body in Arabic and the absence of upper case letters at the beginning of sentences in Arabic. The method presented was found to have an overall correctness of about 60%.

The segmentation free method attempts to perform spotting and segmentation concurrently. Rather than a candidate word image, an entire line image acts as input. The line is split into segments based on an algorithm similar to the ligature-based segmentation algorithm used in [4]. All realistic combinations of adjacent connected components are considered as potential areas where the desired word may appear. This approach more exhaustively searches a line, looking for a given word image, while at the same time keeping the number of evaluations manageable by considering only a small subset of potential regions in the image.

4.2 Automatic Word Segmentation

The process of word segmentation begins with obtaining the set of connected components for each line in the document image. Figure 3 and Figure 4 show the connected components (exterior and interior contours) of a small region of a document image. The interior contours or loops in a component

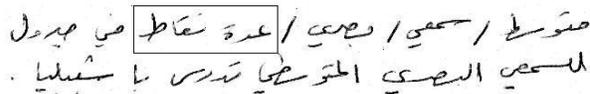


Figure 3: Region of a document image



Figure 4: Connected component exterior and interior contours

are ignored for the purpose of word segmentation as they provide no information for this purpose. The connected components are grouped into clusters, by merging minor components such as dots above and below a major component. Also particular to Arabic, many words start with the Arabic character “Alef”. The presence of an “Alef” is a strong indicator that there may be a word gap between the pair of clusters. The height and width of the component are two parameters used to check if the component is the character “Alef”. Figure 7 shows samples of the Arabic character “Alef”. Every pair of adjacent clusters are candidates for word gaps. Nine features are extracted for these pairs of clusters and a neural network is used to determine if the gap between the pair is a word gap. The nine features are: width of the first cluster, width of second cluster, difference between the bounding box of the two clusters, flag set to 1 or 0 depending on the presence or absence of the Arabic character “Alef” in the first cluster, the same flag for the second cluster, number of components in the first cluster, number of components in the second cluster, minimum distance between the convex hulls enclosing the two clusters and the ratio between, the sum of the areas enclosed by the convex hulls of the individual clusters, to the total area inside the convex hull enclosing the clusters together. The minimum distance between convex hulls is calculated by sampling points on the convex hull for each connected component and calculating the minimum distance of all pairs of such points. Figure 5 and Figure 6 show two pairs of adjacent clusters tested for word gaps. Dotted lines indicate convex hulls around the individual clusters. Solid lines indicate convex hull around the two clusters taken together.

A neural network was trained using these nine fea-

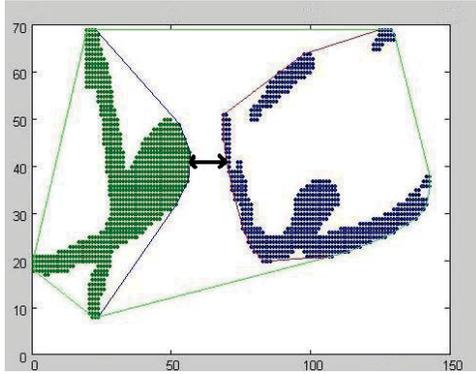


Figure 5: Gap between the convex hulls is not a word gap



Figure 6: The gap between the convex hulls is a word gap

tures with the feature vector labeled as to whether it is a word gap or not. This is similar to the neural network approach used for English postal addresses [5], but with different features.

4.2.1 Word Segmentation Performance

When applied to the document set described earlier with correctly segmented lines, the overall performance is about 60% using a set of seven segmentation features. In [10], the authors noted that a more complex set of features is expected to yield a higher level of performance.

4.3 Segmentation-free Line Processing

The segmentation free algorithm processes the words on a per line basis rather than relying on pre-segmented words. The algorithm can be viewed as a sequence of steps. First, the image is processed into



Figure 7: Samples of Arabic character “Alef”. The height and width are two parameters that are used to detect the presence of “Alef” in the clusters.

component lines. Candidate segmentation points are generated for a given line. The line is scanned with a sliding window, generating candidate words and scoring them, as well as filtering out nearly equivalent candidates.

4.3.1 Segmentation Algorithm

The segmentation algorithm used on the line is essentially the same as the one used to generate candidate character segmentation points in candidate words in the actual spotting step. It is performed via a combination of ligatures and concavity features on an encoded contour of the components of the image. Average stroke width is estimated and used to determine the features.

Ligatures, as noted in [3] are strong candidates for segmentation points in cursive scripts. Ligatures are extracted in a similar way as in [3]—if the distance between y -coordinates of the upper half and lower half of the outer contour for a x -coordinate is less than or equal to the average stroke width, then the x -coordinate is marked as an element of a ligature. Concavity features in upper contour and convexities in the lower contour are also used to generate candidate segmentation points, which are especially useful for distinct characters which are touching, as opposed to being connected. A ligature will cause any overlapped concavity features to be ignored. For a given x -coordinate, if a concavity and convexity overlap, a segmentation point is added for that x -coordinate.

While the character based method described in [3] uses this segmentation method to split a word into candidate characters, the segmentation free line processing method uses it to split the line, the motivation being to generate candidate word regions on the line. Arabic has predictable breaks in a word based on non-connective characters. Therefore, the number of connected components in a word is predictable as well.

4.3.2 Line Scanning

The method utilizes a sliding window, starting from the left of the line and proceeding to the right (in the opposite direction to the way Arabic is written), although the direction of the scan is unimportant because all realistic combinations of connected components will be considered.

Each character class c in Arabic is associated with a minimum and a maximum durational length ($minlen(c)$ and $maxlen(c)$ respectively). These lengths are generated by segmenting a representative dataset of characters with the same segmentation algorithm, and taking the min and max for each character. Due to the nature of the Arabic character set, the upper bound for all characters is 5, not 4 as in [4].

The scanning algorithm will scan for candidate words consisting of a range of segments. For a given search word W of length n , for each character $c_i \in W$, the minimum length $minlen(W)$ considered is $\sum_{i=0}^{n-1} minlen(c_i)$ and the maximum considered length $maxlen(W)$ is $\sum_{i=0}^{n-1} maxlen(c_i)$.

The scanning algorithm starts at each segmentation point p on a line. For a given point p_i , if $i = 0$ or if $p_i.left > p_{i-1}.right$ (i.e., there is horizontal space to the left of the segmentation point) it is considered a valid start point. Similarly, for a given point p_i , if $i = max(p)$ or if $p_i.right < p_{i+1}.left$ (i.e., there is horizontal space to the right of the segmentation point) it is considered a valid endpoint. The algorithm considers candidate words to be ranges of segments between two segmentation points p_i and p_j where p_i is a valid start point, $minlen(W) \leq j - i + 1 \leq maxlen(W)$, and p_j is a valid endpoint.

While this generally results in more candidate words than the other segmentation method, since each Arabic word is only broken into a few pieces separated by whitespace it does not result in a dramatic decrease in performance.

4.3.3 Filtering

Often, a candidate word influences neighboring candidate words' scores. Neighboring candidate words are those words with overlapping segments. Often, a high scoring word will result in high scores for neighbors. The largest issue rises when the high scoring word is, in fact, an incorrect match. In this case, the incorrect choice and several of its neighboring words receive similarly good scores, pushing the rank of the actual word lower in the list. Another issue is if the word being searched for appears multiple times in a document. The best matching words' neighboring candidates depresses the second occurrence's rank. Various ways of dealing with the overlap meet with different degrees of success.

The approach taken in the current incarnation of the algorithm is to keep the candidate word that has the highest score out of the overlapping words. Unfortunately, this occasionally removes the correct word completely from the list. Alternate methods of filtering are being explored.

5 Word Shape Matching

The word segmentation is an indexing step before using the word shape matching for word retrieval. A two-step approach is employed in performing the search: (1) prototype selection: the query (English text) is used to obtain a set of handwritten samples of that word from a known set of writers (these are the prototypes), and (2) word matching: the prototypes are used to spot each occurrence of those



Figure 8: Candidate word regions

words in the indexed document database. A ranking is performed on the entire set of test word images—where the ranking criterion is the mean similarity score between prototype words and the candidate word based on global word shape features.

5.1 Prototype Selection

Prototypes which are handwritten samples of a word are obtained from an indexed (segmented) set of documents. These indexed documents contain the truth (English equivalent) for every word image. Such an indexing can be done using a transcript mapping approach such as described in [2]. Synonymous words if present in the truth are also used to obtain the prototypes. Hence queries such as “country” will result in selecting prototypes that have been truthed as “country” or “nation” etc... A dynamic programming Edit Distance algorithm is used to match the query text with the indexed word image's truth. Those with distance as zero are automatically selected as prototypes. Others can be selected manually.

5.2 Word Matching

The word matching algorithm uses a set of 1024 binary features for the word images. These binary features are compared using the correlation similarity measure 5.2.1 to obtain a similarity value between 0 and 1. This similarity score represents the extent of match between two word images. The smaller the score, the better is the match. For word spotting, every word image in the test set of documents are compared with every selected prototype and a distribution of similarity values is obtained. The distribution of similarity values is replaced by its arithmetic mean. Now every word is sorted in rank in accordance with this final mean score. Figure 9 shows an example query word image compared with the a set of 4 selected prototypes.

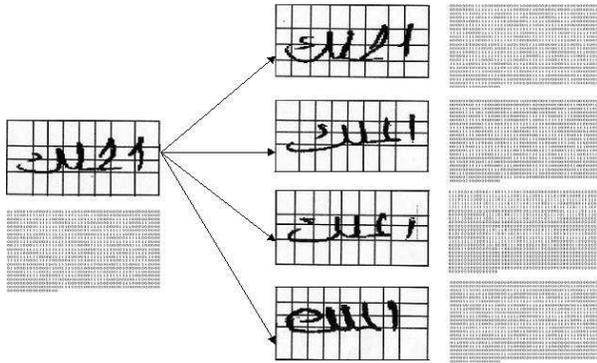


Figure 9: One query word image (left) matched against four selected prototype images. The 1024 binary GSC features are shown next to the images.

5.2.1 Similarity measure

The method of measuring the similarity or distance between two binary vectors is essential. The correlation distance performed best for GSC binary features [14] which is defined for two binary vectors X and Y , as in equation 1

$$d(X, Y) = \frac{1}{2} \left(1 - \frac{s_{11}s_{00} - s_{10}s_{01}}{[(s_{10} + s_{11})(s_{01} + s_{00})(s_{11} + s_{01})(s_{00} + s_{10})]^{1/2}} \right) \quad (1)$$

where s_{ij} represent the number of corresponding bits of X and Y that have values i and j .

5.3 Word-Matching Performance

The performance of the word spotter was evaluated using manually segmented Arabic documents. All experiments and results were averaged over 150 queries. For each query, a certain set of documents are used as training to provide for the template word images and rest of the documents were used for testing. Figure 14 shows the percentage of times when correct word images were retrieved within the top ranks scaled on the x-axis. More than one correct word image does exist in the test set. For example, if the word “king” occurs in document 1 twice, and we use writers 1 through 6 for training, then we have $6 * 2 = 12$ template word images to perform word spotting. The remaining test set contains $(10 - 6) * 2 = 8$ correct matches in the test set. The top curve of the figure shows that at least one correct word is retrieved 90% of the time within the top 12 ranks. The other curves show the same when at least 2 and 3 correct words are retrieved. Similar plots can be obtained when different numbers of writers are used for training.

Performance of word spotting can be specified in terms of precision and recall. If n_1 is the number of relevant words in the database, n_2 is the number of words retrieved, and n_3 is number of relevant words among those that are retrieved then $Precision = n_3/n_2$ and $Recall = n_3/n_1$.

6 Text String Matching

The approach is lexicon based; that is, it makes use of the Arabic sequences of characters for words in the lexicon in order to select sets of prototype images representing the characters forming them. A preprocessing step is a line and word segmentation process, such as the one described in [10]. The candidate word image is first split into a sequence of segments (as in Figure 10), with the ideal result being individual characters in the candidate word being separated. The segmentation algorithm used “over-segments” words in the hopes of avoiding incorrectly putting more than a single character into a segment. Segments are then rejoined and features extracted, which are in turn compared to features of prototype images of the characters. Further issues of undersegmenting unique to Arabic are dealt with using compound character classes. A score that represents the match between the lexicon and the candidate word image is then computed. The score relates to the individual character recognition scores for each of the combined segments of the word image.



Figure 10: Arabic word “king”, with segmentation points and corresponding best segments shown below

6.1 Character Dataset

Unlike the method presented in [10], this method depends on character images rather than word images to form a basis for comparison. In this method, the image is essentially cut into component characters and each character is matched for similarity, in contrast to the latter’s method of matching entire word shape. Since the number of characters in Arabic is a rather small compared with the number of potential words, a library of component characters can be incorporated directly into the system. This eliminates the indexing phase of [10].

As such a character database was not readily available, a new character image dataset was derived from the existing Arabic document dataset produced from the CEDARABIC [9] project. The original dataset consisted of a collection of handwritten doc-

uments produced from a variety of authors and is described in Section 7.1. The scanned words were individualized and raw ASCII descriptions of the Arabic characters were given corresponding to the content of the word. The derived dataset consists of images of single Arabic characters and character combinations. Approximately 2,000 images of characters and character combinations in other configurations were created by allowing the ligature based segmentation algorithm to create candidate supersegments of the truthed words, and manually matching the best candidate supersegments to the corresponding character or character combination when the segmentation was successful. Both left to right and right to left versions of the writings were tested, the original right to left images producing better results (left to right versions occasionally seemed to be more prone to undersegmenting the words). The 2,000 images represent but a small fraction of potential images from this dataset. Work on extending this dataset is ongoing.

6.2 Features

WMR features for each of the character images were extracted and incorporated into the recognition engine of CEDARABIC. As described in [11], the WMR feature set consist of 74 features. Two are global features—aspect and stroke ratio of the entire character. The remaining 72 are local features. Each character image is divided into 9 subimages. The distribution of the 8 directional slopes for each subimage form this set (8 directional slopes \times 9 subimages = 72 features). $F_{l_{i,j}} = s_{i,j}/N_i S_j$, $i = 1, 2, \dots, 9$, $j = 0, 1, \dots, 7$, where $s_{i,j}$ = number of components with slope j from subimage i , where N_i = number of components from subimage i , and $S_j = \max(s_{i,j}/N_i)$. These features are the basis of comparison for the character images derived from the segmentation of words to be recognized. To date, this appears to be the first application of WMR features to Arabic recognition.

To obtain preliminary results, the base shape of a letter was mapped to all derivations of that letter. For example, the base shape of the character *beh* was mapped to *beh*, *teh*, and *theh*. The initial and medial forms of *beh* were also mapped to the initial and medial forms of *noon* and *yeh*. If separately truthed versions were available specifying explicit membership to, for example, *teh*, such characters were only included in *teh*'s set of features.

6.3 Image Processing And Segmentation

Image processing is accomplished via a method similar in part to that described in [3]. First, a chain code representation of the binary image's contours

is generated. Noise removal, slant correction, and smoothing is performed. Segmentation is performed via a combination of ligatures and concavity features on an encoded contour of the components of the image. Average stroke width is estimated and used to determine the features. The number of segmentation points is kept to a minimum, but unlike in [3], the maximum number of segmentation points per character is five. WMR features are extracted from segments.

One goal of this method is to oversegment words with the hopes of eliminating undersegmentation altogether. Undersegmentation in ligature based segmentation of Arabic text, however, continues to be problematic due to the presence of character combinations and vertically separated characters. For example, some writing styles do not mark certain letters with much clarity—especially initial characters, for example initial *yeh*'s. Since the ligature based segmentation proceeds horizontally, seeking breaking points at various positions along the x -axis, the vertical “stacking” of characters cannot be dealt with simply by increasing the sensitivity of the segmentation (see Figure 11). To deal with these issues, character classes were defined corresponding to the common character and vertically occurring combinations.



Figure 11: Left: (horizontally) unsegmentable character combination, Right: individual images

6.3.1 Preprocessing Lexicon

An Arabic word is specified as a sequence of the approximately 28 base letters. To aid recognition, a simple algorithm maps the given text to the correct variation of each character. For example, “Alef|Lam|Teh|Qaf|Alef maksura|” is mapped to “Alef_i|Lam_i|Teh_m|Qaf_m|Alef maksura_f|” where “i” means the letter is in the initial position, “m” means the letter is in the medial position, etc. Additional post-processing steps to the Arabic lexicon combine adjacent individual characters in appropriate positions into character combination classes. For example, “Lam_i|Meem_m|” is mapped to “Lammeem_i|.”

The new mapping system for the 150 new character classes were incorporated into the character recognition model of CEDARABIC, replacing the support for English letters carried over from CEDAR-FOX.

6.3.2 Word Recognition

The objective is to find the best match between the lexicon and the image. In contrast to [3], up to five adjacent segments are compared to the character classes dictated as possibilities by a given lexicon entry. In the first phase of the match, the minimum Euclidean distance between the WMR features of candidate supersegments and the prototype character images is computed. In the second phase, a global optimum path is obtained using dynamic programming based on the saved minimum distances obtained in the first matching phase. The lexicon is ranked, the entries with the lowest total scores being the closest matches. Figure 10 shows the the Arabic word “king” split into its best possible segments.

Testing proceeded on the same 10 authors’ documents as in [10]. Recognition was attempted on approximately 180 words written by each of the 10 authors (for a total of approximately 1,800 words). Recognition was attempted in two runs, one with a lexicon size of 20 words and one with a size of 100. The lexicon was generated from other words among the 180 being recognized. The words and the lexicons in the tests were the same for all authors.

6.4 Word Spotting

Word spotting proceeds in a very similar fashion to word recognition. In the case of word spotting, the lexicon consists only of the word being spotted. A score against this lexicon entry is generated for each candidate word in the document. The candidate words are ranked according to score, the words with the best scores are most likely to be the word being spotted.

From the documents written by the authors, 32 words were chosen at random and “spotted,” in a similar fashion to the experiments performed in [10]. Note that the recall for word spotting when utilizing the expanded Arabic character classes is nearly 80% for a precision of 50, which is a significant improvement over the other methods (using simply the Arabic letters individually and the image based method described in [10]).

7 Results

7.1 Document Image Database

For evaluating the results of our methods, we used a document collection prepared from 10 different writers, each contributing 10 different full page documents in handwritten Arabic. Each document is comprised of approximately 150 – 200 words each, with a total of 20, 000 word images in the entire database. The documents were scanned at a resolution of 300 dots per inch. This is the resolution to be used for optimal performance of the system. Fig-

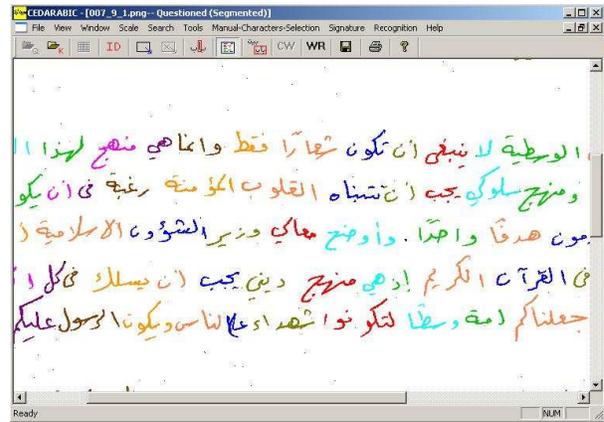


Figure 12: Sample “.arb” file opened with CEDARAR-BIC. Each segmented word is colored differently from adjacent words.

ure 12 shows a sample scanned handwritten Arabic document written by writer 1.

For each of the 10 documents that were handwritten, a complete set of truth values, comprised of the alphabet sequence, meaning, and the pronunciation in that document was also given. The scanned handwritten documents’ word images were mapped with the corresponding truth information (see Figure 13).

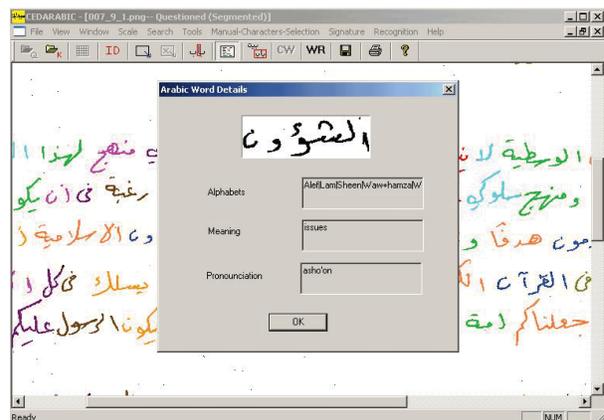


Figure 13: Word image with corresponding truth

7.2 Experiments

To test the combined method, 32 queries were issued on 13,631 records. The neural network was trained on 300 such queries, 150 positive and 150 negative query matches. All remaining records were used for testing. The combined score is a score between -1 and 1. A negative score would indicate a mismatch and a positive score a match. A 91% raw classification accuracy was observed.

Using five writers for providing prototypes and the other five for testing, using manually segmented documents, 55% precision is obtained at 50% recall

for the word shape method alone. The character based method achieves 75% precision at the same recall rate. The combined method is consistently better, resulting in about 80% precision. A comparison graph, with the word shape method using five writers, is shown in Figure 14. One search result from CEDARABIC is shown in Figure 15.

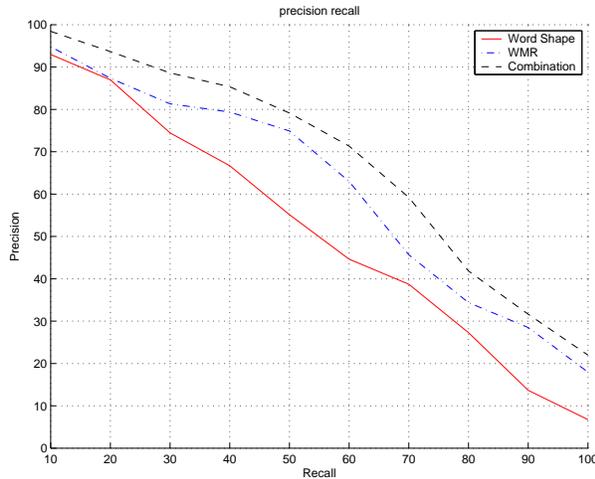


Figure 14: Precision-recall comparison of word-shape, character-shape and combination approaches.

8 Conclusion and Future Directions

Processing image and text based queries in parallel can result in higher performance than either alone. The versatile search framework presented can be applied to many document search problems. The example presented illustrates a word spotting application, but other document search strategies may experience similar performance increases. For example, a partially transcribed document could be represented as a “bag of words,” with a term/document matrix. From there, latent semantic analysis TF-IDF (term frequency-inverse document frequency)

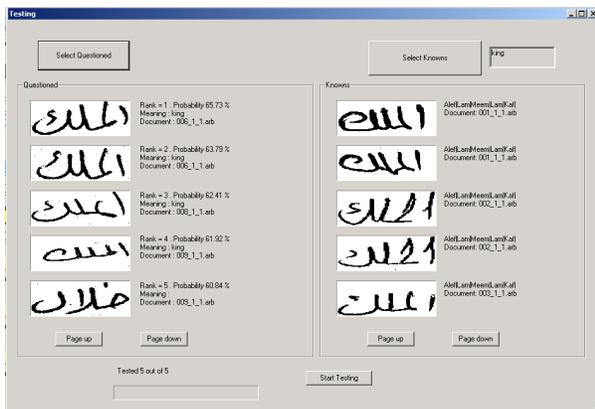


Figure 15: Word Spotting Testing Results

weights can be used to perform traditional searches, with image search augmenting less than perfect transcription techniques. Furthermore, “plugging in” improved image or text-based search algorithms can push overall performance higher. For our experiments, we used the neural network to simply weight the two incoming scores. However, features extracted from the images may improve neural network performance.

References

- [1] D. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining*. MIT Press, Cambridge, MA, 2001.
- [2] C. Huang and S. N. Srihari. Mapping transcripts to handwritten text. In *Proc. Tenth International Workshop on Frontiers in Handwriting Recognition (IWFHR)*, La Boule, France, 2006. IEEE Computer Society.
- [3] G. Kim. *Recognition of offline handwritten words and extension to phrase recognition*. Doctoral Dissertation, State University of New York at Buffalo, 1997.
- [4] G. Kim and V. Govindaraju. A lexicon driven approach to handwritten word recognition for real-time applications. In *IEEE Transactions on Pattern Analysis and Machine Intelligence 19(4)*, pages 366–379, 1997.
- [5] G. Kim, V. Govindaraju, and S. N. Srihari. A segmentation and recognition strategy for handwritten phrases. In *International Conference on Pattern Recognition, ICPR-13*, pages 510–514, 1996.
- [6] G. Russell, M.P. Perrone, Yi min Chee, and Airnan Ziq. Handwritten Document Retrieval. In *Proc. Eighth International Workshop on Frontiers in Handwriting Recognition*, pages 233–238, Niagara-on-the-lake, Ontario, 2002.
- [7] S. N. Srihari, C. Huang, and H. Srinivasan. A search engine for handwritten documents. In *Document Recognition and Retrieval XII: Proceedings SPIE*, pages 66–75, San Jose, CA, 2005.
- [8] S. N. Srihari and Z. Shi. Forensic handwritten document retrieval system. In *Proc. Document Image Analysis for Libraries (DIAL)*, pages 188–194, Palo Alto, CA, 2004. IEEE Computer Society.
- [9] S. N. Srihari, H. Srinivasan, P. Babu, and C. Bhole. Handwritten Arabic word spotting

using the CEDARABIC document analysis system. In *Proc. Symposium on Document Image Understanding Technology (SDIUT-05)*, pages 123–132, College Park, MD, Nov. 2005.

- [10] S. N. Srihari, H. Srinivasan, P. Babu, and C. Bhole. Spotting words in handwritten Arabic documents. In *Document Recognition and Retrieval XIII: Proceedings SPIE*, pages 606702–1 to 606702–12, San Jose, CA, 2006.
- [11] S. N. Srihari, C. I. Tomai, B. Zhang, and S. Lee. Individuality of numerals. In *Proc. Seventh International Conference on Document Analysis and Recognition (ICDAR)*, page 1096, Edinburgh, UK, 2003. IEEE Computer Society.
- [12] S. N. Srihari, B. Zhang, C. Tomai, S. Lee, Z. Shi, and Y-C. Shin. A search engine for handwritten documents. In *Proc. Symposium on Document Image Understanding Technology (SDIUT-05)*, pages 67–75, Greenbelt, MD, 2003.
- [13] K. Taghva, J. Borsack, and A. Condit. Results of applying probabilistic IR to OCR text. In *Research and Development in Information Retrieval*, pages 202–211, 1994.
- [14] B. Zhang and S. N. Srihari. Binary vector dissimilarity measures for handwriting identification. *Proceedings of the SPIE, Document Recognition and Retrieval*, pages 155–166, 2003.