A Semi-automatic Adaptive OCR for Digital Libraries

Sachin Rawat, K.S. Sesh Kumar, Million Meshesha, Indraneel Deb Sikdar,
A. Balasubramanian, and C.V. Jawahar

Centre for Visual Information Technology,
International Institute of Information Technology, Hyderabad - 500032, India
jawahar@iiit.ac.in

Abstract. This paper presents a novel approach for designing a semi-automatic adaptive OCR for large document image collections in digital libraries. We describe an interactive system for continuous improvement of the results of the OCR. In this paper a semi-automatic and adaptive system is implemented. Applicability of our design for the recognition of Indian Languages is demonstrated. Recognition errors are used to train the OCR again so that it adapts and learns for improving its accuracy. Limited human intervention is allowed for evaluating the output of the system and take corrective actions during the recognition process.

1 Introduction

It is becoming increasingly important to have information available in digital format for effective access, reliable storage and long term preservation [1, 2, 3]. This is catalysed by the advancement of Information Technology and the emergence of large digital libraries for archival of paper documents. One of the projects aimed at large scale archival is Digital Library of India (DLI) [4]. The DLI aims at digitizing all literary, artistic, and scientific works of mankind so as to provide better access to traditional materials and make documents freely accessible to the global society.

Efforts are also exerted to make available the content of these digital libraries to users through indexing and retrieval of relevant documents from large collections of document images [5]. Success of document image indexing and retrieval mainly depends on the availability of optical character recognition systems (OCRs). The OCR systems take scanned images of paper documents as input, and automatically convert them into digital format for on-line data processing. The potential of OCRs for data entry application is obvious: it offers a faster, highly automated, and presumably less expensive alternative to the manual data entry process. Thus they improve the accuracy and speed in transcribing data to be stored in a computer system [1]. High accuracy OCR systems are reported for English with excellent performance in presence of printing variations and document degradation. For Indian and many other oriental languages, OCR systems are not yet able to successfully recognise printed document images of varying scripts, quality, size, style and font. Therefore the Indian language documents in digital libraries are not accessible by their content.

In this paper we present a novel approach for designing a semi-automatic and adaptive OCR for large collection of document images, focusing on applications in digital libraries. The design and implementation of the system has been guided by the following facts:

H. Bunke and A.L. Spitz (Eds.): DAS 2006, LNCS 3872, pp. 13–24, 2006.

To work on diverse collections: It has been realised that, we need to design an OCR system that can register an acceptable accuracy rate so as to facilitate effective access to relevant documents from these large collection of document images. Acceptability of the output of the OCR system depends on the given application area. For digital libraries, it is desirable to have an OCR system with around 90% accuracy rate on 90% of the documents rather than having recognition rate of 99% on 1% of the document collections.

Possibility of human intervention: The main flow of operations of the digitization process we follow in the Digital Library of India is presented in Figure 1. In projects like DLI, there is considerable amount of human intervention for acquiring, scanning, processing and web enabling the documents. The expectation of a fully automatic OCR is impractical at the present situation due to the technological limitations. We exploit the limited manual intervention present in the process to refine the recognition system.

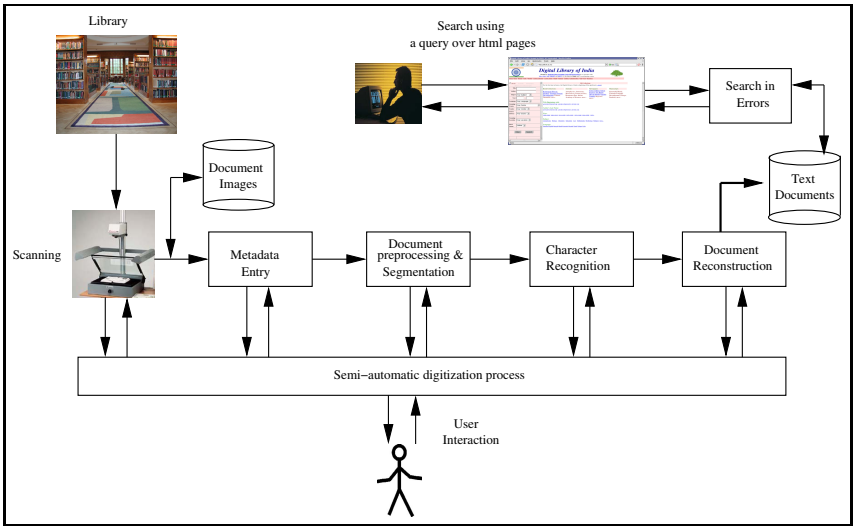

Fig. 1. Block Diagram of the Flow of Operations in Content Generation and Delivery in Digital Libraries

Effectiveness of GUI: There has been considerable amount of research in Indian language OCR [6]. There are many modules in the OCR and most of them are sequential in nature. Failure of even one of the modules can result in making the system unacceptable for real-life applications. Commercial organizations have not taken enough interest to comeup with an Indian language OCR. We argue that, with effective user interfaces to correct possible errors in one or more modules, one can build practical OCR systems for Digital Library of India pages. This can avoid the failure of the entire system due to the failure of selected blocks (say script separation or text-graphics identification).

Availability of limited research: Even with the presence of the first OCR in 1960, English did not have an omni-font recognition system till 1985 [7]. Most of the research in

Table 1. Diversity of Document Collections in DLI. They Vary in Languages, Fonts, Styles and Sizes. Books that are Published since 1850 are Archived which have Different Typesets.

Documents	Diversity of Books
Languages	Hindi, Telugu, Urdu, Kannada, Sanskrit, English, Persian, European, others
Typesets	Letter press, Offset printer, Typewriter, Computer Typeset, Handwritten
Fonts	Ranging from 10 to 20 fonts in each language
Styles	Italics, Bold, Sans serifs, Underline, etc.
Sizes	8 to 45 points
Year of Publication	Printed books from 19, 20 and 21st centuries Ancient manuscripts to 2005

OCR technology has been centered around building fully automatic and high performing intelligent classification systems. Summary of the diverse collections of documents in Digital Library of India is presented in Table 1. Building an omnifont OCR that can convert all these documents into text does not look imminent. At present, it may be better to design semi-automatic but adaptable systems for the recognition problem.

Adaptability requirements: There are many excellent attempts in building robust document analysis systems in industry, academia and research institutions [8, 9]. None of these systems are applied or tested on Indian language real-life documents. Most of the automatic OCR systems were trained offline and used online without any feedback. We argue for a design which aim at training the OCR on-the-fly using knowledge derived from an input sequence of word images and minimal prior information. The resulting system is expected to be specialized to the characteristics of symbol shape, context, and noise that are present in a collection of images in our corpus, and thus should achieve higher accuracy. This scheme enables us to build a generic OCR that has a minimal core, and leave most of the sophisticated tuning to on-line learning. This can be a recurrent process involving frequent feedbacks and refinement. Such a strategy is valuable in situations like ours where there is a huge collection of degraded and diverse documents in different languages [10].

2 Role of Adaptation and Feedback

Recognition of scanned document images using OCR is now generally considered to be a solved problem [11]. Many commercial packages are available for Latin scripts. These packages do an impressive job on high quality originals with accuracy rates in the high nineties. However, several external issues can conspire against attaining this rate consistently across various documents, which is necessary for digital libraries.

Quality: The quality of the original document greatly affects the performance of an OCR system. This is because (i) the original document is old, and has suffered physical degradation, (ii) the original is a low quality document, with variations in toner density, and (iii) many characters are broken which presumably are faint areas on the original not picked up by the scanning process. For example, segmentation errors may occur because the original document contains broken, touching and overlapping characters.

Scanning Issues: Degradations during scanning can be caused by poor paper/print quality of the original document, poor scanning equipment, etc. These artifacts lead to recognition errors in other stages of the conversion process.

Diversity of Languages: Digital libraries archive documents that are written in different languages. Some languages (such as Hindi and Urdu) have complex scripts, while others (like Telugu) have large number of characters. These are additional challenges for the OCRs design.

Any of the above factors can contribute to have insufficient OCR. We need to consider implementation issues and design effective algorithms to address them. Algorithms used at each stage of the recognition for preprocessing, segmentation, feature extraction and classifications affects the accuracy rate. We should allow users to inspect the results of the system at each stage. If user can select the appropriate algorithm or set the most apt parameters, performance of the individual modules and thereby that of the system can be maximised. Designing a system that supports an interactive processing is therefore crucial for better recognition result. We claim that application of a semi-automatic adaptive OCR has a great role in this respect. This will further create dynamism to update existing knowledge of the system, whenever new instances are presented during the recognition process.

2.1 Role of Postprocessor

A post processor is used to resolve any discrepancies seen between recognized text and the original one. It significantly improves the accuracy of the OCR by suggesting, often, a best-fit alternative to replace the mis-spellings. For this, we use a reverse dictionary approach with the help of a trie data structure. In this approach, we create two dictionaries. One is filled with words in the normal manner and the other with the same words reversed. We narrow down the possible set of choices by using both dictionaries. It interacts with the OCR in the following way.

- Get the word and its alternative(s) (if suggested by the OCR).
- Check which of them form valid words and then based on their weights choose the optimal one.
- In case of failure, the postprocessor suggests its own list of alternative words.
- Check with the OCR whether its own suggestion is visually similar with the original one.

In this way, the postprocessor correct recognition errors. Due to the errors in segmentation or classification, the recognition module can have three categories of errors: dele-

Table 2. Performance of the Post Processor on Malayalam Language Datasets. Malayalam is One of the Indian Language with Its Own Scripts.

Error Type	Deletion	Insertion	Substitution
% of errors corrected	78.15	58.39	92.85

tion, insertion and substitution. Table 2 shows percentage of errors corrected by our Malayalam post-processor.

The key issue in document-specific training is that the transcription may be limited or may not be available *a priori*. We need to design an adaptive OCR that learns in presence of data uncertainty and noise. In the case of training, this means that the training transcriptions may be erroneous or incomplete. Thus, one way to produce the required training data is to use words analyzed by the post processor. We design this architecture explicitly, such that the postprocessor accepts words generated by an OCR system and produces error-corrected ones. These words, identified by the postprocessor, are fed back to the OCR such that it builds its knowledge through training for improvement of its performance.

2.2 Role of User Feedback

Designing a mechanism for feedback enables the system to obtain corrective measures on its performance. Users interact with the system and then investigate the output of the system. Based on the accuracy level of the OCR, they will communicate to the system those words in error. This will enable the OCR to gain additional knowledge, based on which retraining is carried out. This mechanism is possible when there is feedback. Thus the feedback will serve as a communication line to obtain more training data that helps to overcome any performance lacuna. This is supported by a GUI which enables to provide feedback to the system concerning a list of words that need corrective action with their full information.

3 Design Novelty of the System

We have a prototype interactive multilingual OCR (IMOCR) system, with a framework for the creation, adaptation and use of structured document analysis applications. IMOCR is a self-learning system that improves its recognition accuracy through knowledge derived in the course of semi-automatic feedback, adaptation and learning. There are various notable goals towards the design and implementation of IMOCR.

- To setup a system that learns through a recurrent process involving frequent feedback and backtracking.
- To manage diverse collection of documents that vary in scripts, fonts, sizes, styles and document quality.
- To have flexible application framework that allows researchers and developers to create independent modules and algorithms rather than to attempt to create an all encompassing monolithic application.
- To allow end users the flexibility to customize the application by providing the opportunity to configure the various modules used, their specific algorithm and parameters.

By doing so the system will be able to interactively learn and adapt for a particular document type. In essence, it provides a framework for an interactive $test \Rightarrow feedback \Rightarrow learn \Rightarrow adapt \Rightarrow automate$ cycle.

3.1 Data Corpus

Our OCR system has been designed under the assumption that vast amount of training samples are not available for every category of document. Rather, it builds a huge collection (or corpus) over time based on the feedback mechanism. The corpus encompass all types of data that are involved in printed documents. The collection contains synthetic data (that varies in scripts, fonts, sizes and styles) and real-life documents (such as books, magazines and newspapers). Having such wealth of data enables the system to learn and accumulate knowledge out of it.

3.2 Design Considerations

From the implementation point of view, the following design considerations have been made:

- The system supports multiple languages in the same framework. Languages currently supported include English, Hindi, Tamil, Telugu and Malayalam.
- Each aspect, right from the GUI to the font-selection are platform independent.
- Dual mode running of the system is enabled by interactive interface as well as batch mode support.
- The application is designed to allow the user to choose the run-time configuration of the system right from the modules, the inner-lying algorithms, the input-output format, the parameters etc. via support for user specified configuration files. The GUI is designed for multi-level access to configuration parameters that scale up for users with various levels of skill.
- A multi-core approach is taken so as to ease development and integration of modules in the future.

3.3 Architecture of the IMOCR

We design the general architecture of an interactive multilingual OCR (IMOCR) system that is open for learning and adaptation. An overview of the architecture of the system is shown in Figure 2. The IMOCR design is based on a multi-core approach. At the heart of the same is an application tier, which acts as the interface between the Graphical User Interface (GUI) and the OCR modules. This application layer identifies the user-made choices, initialises data and document structures and invokes relevant modules with suitable parameters. The GUI layer provides the user with the tools to configure the data-flow, select the plug-ins and supply initialization parameters. The system provides appropriate performance metrics in the form of graphs and tables for better visualization of the results of each step and module during the recognition process.

The last layer is the module/algorithm layer where the actual OCR operations are done. This layer is segmented based on clearly identified functionality. Each module implements a standard interface to be invoked via the application. Each module internally can decide on multiple algorithm implementations of the same functionality that may be interchanged at run-time. This helps in selection and use of an appropriate algorithm or a set of parameters for a book, collection or script. This layer is designed on the principle of plug-ins. The system allows transparent runtime addition and selection

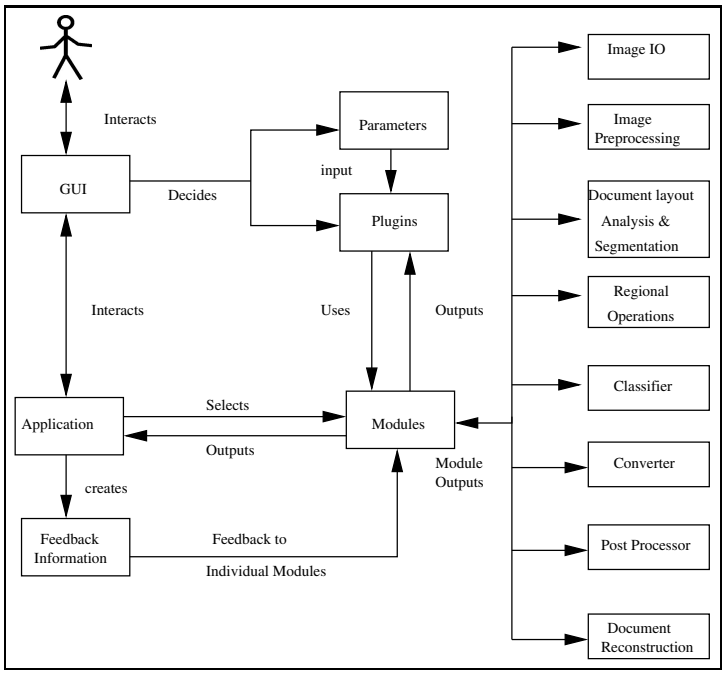

Fig. 2. An Architecture of the Prototype OCR System

of modules (as shared objects/ dynamic libraries) thereby enabling the decoupling of the application and the plug-ins. Feature addition and deployment is as simple as copying the plug-in to the appropriate directory. The other advantages of this approach are lower size of application binary, lower runtime memory footprint and effective memory management (through dynamic loading and unloading, caching etc.).

4 Recognition Module

The recognition module has three main parts: the segmenter, the labeler and the resolver. Scanned images are initially preprocessed to improve the performance of the recognition process [12]. They are binarized and skew corrected. We implement Isodata for binarization and Yin's algorithms [13] for skew detection and correction. The segmentation module is designed to analyze the overall structure of the document and identify sub-regions such as text, tables, images/graphics etc.

Text blocks are further segmented into lines and words. There are several layout analysis algorithms available in literature. The process of segmentation of a page is carried out sequentially in two major steps. The first is separation of text and image regions. Then, the segmentation of text into columns, blocks, text lines and words. The performance of the page layout analysis module depends on the type of document image. Every algorithm has a set of freely tunable parameters, which can be set to give apt results for the document image. However, these parameters are either set by interacting

with the human or automatically by using a learning algorithm [14]. In some cases, the output of the layout analysis may not give apt results for any parameter value. We provide a mechanism for the human to correct the segmented output for further analysis of the document. These human corrections can inturn be used to improve the segmentation process over iterations.

The segmenter comes up with a list of words that are input to the labeler for feature extraction and training. The labeler detects characters and their constituent components using connected component analysis and extracts features of connected components using principal component analysis (PCA). There are a large number of classes available in the Indian Languages. Therefore there is a large amount of training overhead in computation and storage. This training overhead is reduced by performing dimensionality reduction using principal component analysis (PCA).

The features in the Eigenspace are used for classification using a Support Vector Machine (SVM). Support Vector Machines have good generalization capability and perform well over Indian language datasets. The labeler produces a vector of class labels that is used to produce the final recognition results. Details of the feature extraction algorithm and the classifier can be found in Jawahar et el. [15].

The resolver module is designed to resolve confusing characters during the recognition process. It uses language specific information to resolve the confusion. Once the confusions are resolved the class labels of each word are converted into UNICODE using a converter. The postprocessor is based on contextual information and

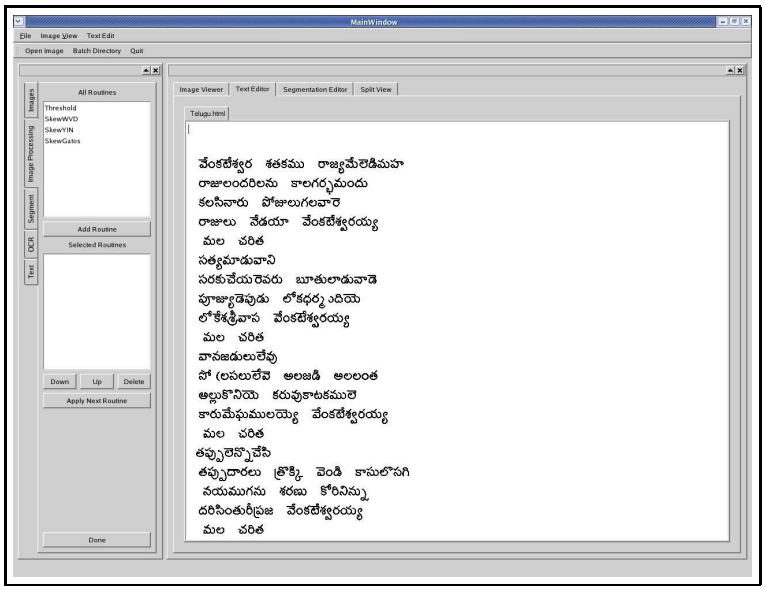

Fig. 3. A Screenshot of Our Prototype Interactive and Adaptive OCR System. The User Interacts for Selection of Algorithms and Parameters, Feedback and Error Correction, Adaptation and Learning.

language-based reverse dictionary approach (as described in Section 2) to correct any mis-recognized words. The post processor can significantly improve the performance of the system. We use a bootstrapping process to retrain the system using the errors corrected by the resolver. The system continously learns through semi-automatic adaptation from feedback; each time gaining knowledge that improves its recognition rate. The feedback is initiated from the result of the resolver.

The output of the postprocessor is a UNICODE string, which can be converted into any language specific font using the converter available. The recognized text is finally reconstructed in various formats including PDF, DOC, RTF, HTML, and Plain text. The output is structurally and visually similar to the input document image as shown in Figure 3.

5 Provision for Learning

In this paper, we have focused on design of a system with capabilities for learning and adaptation. In the previous sections, we described the design criteria to suite these functionalities and the implementation techniques. Improvement in the system, as of now, is limited to the recognition. We also plan to adapt our segmentation work into this framework [14]. Adaptation is primarily done to improve the performance of the system on a specific collection, say a book. After interactively digitizing a few pages, a user can ask the system to retrain and improve the classification performance for the rest of the pages. During retraining, the OCRed words are analysed against the post-processed words to determine the existence of recognition errors. If there is an error, these words are candidates for retraining. To this effect, information related to each word (such as language, font, location indicator coordinates, etc.) are fedback to to the system so that each of its components are added to their respective classes. Finally the system is trained again using the updated datasets to create a new model. This process can repeat multiple times until a saturation (or even over-training) happens on the collections.

An intelligent system should be capable of doing more than adaptation. It needs to identify the appropriate parameters and algorithms without any user interaction and learn to perform better over time. This needs many more functionalities in the system design and implementation. A postprocessor (critique) may run in three modes – student (when a newly seen word is added to the dictionary), mentor (when suggestions are made to the user), and teacher (when the word is replaced in the document). Role of the postprocessor may be set from the user interface or automatically learnt from the confidence level of the recogniser. Similarly, the entire system may want to identify reliable examples for getting trained or learnt from the results. System may use the character (component) images for retraining/refinement depending on the configuration and mode of the system.

The improvement of the performance is presently done by retraining the modules. However, a better alternative will be to refine the performance, provided the algorithms support incremental modes of learning. In our design, the implementation is done such that the retraining or refinement process is transparent to the user. Individual modules can operate in either mode depending on the user setting or algorithmic limitations.

In addition to the adaptation, we have provided provision for long term learning and performance improvement. Adaptation is usually local in nature and is very effective for a specific collection. However, for a new collection, if one can identify a suitable set of model parameters (algorithms), it would further reduce the manual intervention. Such situations are frequent in digital libraries. The settings (parameters) used for one of the books could be an acceptable choice for most books from the same publisher or printed in the same press. We propose to identify the most suited available parameters by modeling the recogniser as a mixture model and selecting the most likely distribution.

6 Performance Analysis of the Prototype System

We have a prototype system that is used for the recognition of document images. The interface is a gate-way to the interactive character recognition. It is an easy to use UI with support for (i) semi automatic operation, (ii) selection of intended regions of image for processing, (iii) display all extracted information (for example, order of segmentation, script, confidence measure, class label, UNICODE etc), (iv) image display at variable resolution, (v) allow for classifier training and display correspondences between image and text.

The system accepts scanned document images in multiple formats such as PGX (PGM, PNM, PPM), TIFF, etc. Scanned images are preprocessed for binarization and skew correction, and then segmented into words. The labeler split words into components and assigned class labels based on which the recognition is performed. Then features are extracted and SVM classifier is used for classification. Algorithms used at each steps for character recognition are already reported in [15, 16]. Recognition results are postprocessed to resolve ambiguity among the candidate words, which is later fedback to the system for creating models that encompasses the new training datasets. The result of the system is reconstructed (see Figure 3) and available in multiple formats such as plain text, HTML, RTF, DOC, PDF etc. The result of the postprocessor is also used for building training data for learning. Our system is semi- automatic for adaptation. Users can have minimal role in due course of the recognition process.

We evaluate the performance of the prototype system on datasets taken from many Indian languages: Hindi, Telugu, Tamil and Malayalam. These datasets are of high quality and used for prototyping and bootstrapping. They are useful for building a pre-

Table 3. Accuracies of DDAG-SVM on Datasets of Various Indian Languages

Name of Language	Number of classes	Accuracy	
		Normal Data	Degraded Data
Hindi	116	98.33	96.54
Telugu	347	97.15	93.14
Tamil	120	99.43	96.24
Malayalam	127	99.72	97.30

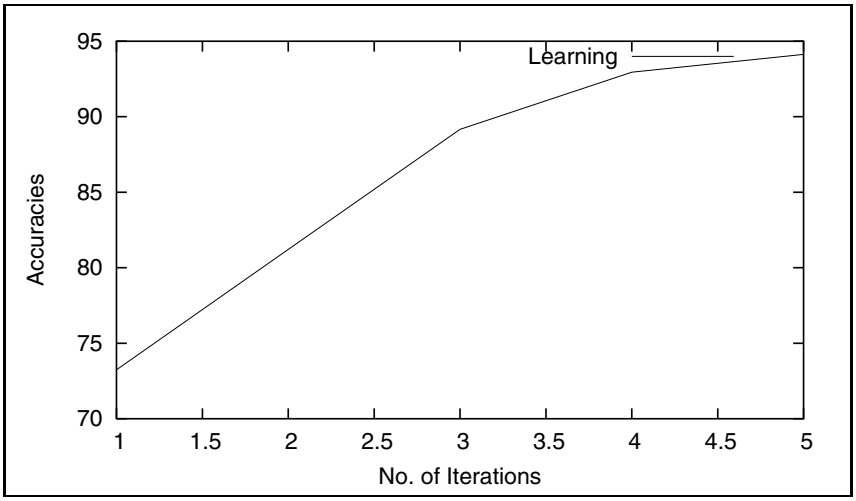

Fig. 4. An Improvement in the Performance of the System Through Learning. It shows Accuracy Vs. Number of Iterations.

liminary version of the training corpus. We add noise using a degradation model as suggested in Kanungo et al. [17]. This degradation simulates the distortions due to imperfection in scanning. The border pixels are eroded using a probabilistic model based on distance from the boundary. We test the system using synthetic data (normal and degraded). The result is shown in Table 3. Such results at the component level are available in many Indian language OCR papers [6]. The errors rates at character level and word level are considerably larger than this. Also the performance degrades highly with the degradation of the input documents.

We now demonstrate how the poor performance on real-life document can be improved using the adaptation strategy discussed in the previous sections. For the first few pages of the book, the accuracy was found to be very low (around 75%) and within few iterations (retrainings) the accuracy improved to close to 95%. This increase is applicable for the specific book, but not generalizable across collections. During adaptation, additional samples of confusing components are added to the training datasets for building improved classifier models. Figure 4 graphically presents the results of the adaptation. It is observed that adapting the system to a maximum of 10 pages is enough for the recognition of an average book with 300 pages. This further indicates how fast the system generalize based on few learnt training data.

7 Conclusions

We have presented a semi-automatic adaptive OCR system. This strategy is important for the recognition of large digitized manuscript datasets of Indian and other oriental languages. The system learns from feedback propagated back through the adaptation process. We claim that interactive OCR lead to better results and is necessary for poor

(or mediocre) quality documents. Performance analysis shows that the system can effectively adapt to documents archived in digital libraries. We are working on advanced learning procedures that automatically interacts and pass feedback back to the system through which it adapts to a given situation easily with few or no human intervention.

Acknowledgment. This work was partially supported by the MCIT, Government of India for Digital Libraries Activities.

References

1. Nagy, G.: Twenty Years of Document Image Analysis in PAMI. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 38–62
2. Pavlidisi, T., Mori, S.: Optical Character Recognition. Proceedings of the IEEE **80** (1992) 1026–1028
3. O. D. Trier, A. K. Jain, T. Taxt: Feature Extraction Methods for Character Recognition: A Survey. Pattern Recognition **29** (1996) 641–662
4. Digital Library of India. (at: `http://www.dli.gov.in`)
5. Doermann, D.: The Indexing and Retrieval of Document Images: A Survey. Computer Vision and Image Understanding (CVIU) **70** (1998) 287–298
6. U Pal, B B Chaudhuri: Indian Script Character Recognition: A Survey. Pattern Recognition **37** (2004) 1887–1899
7. Kahan, S., Pavlidis, T., Baird, H.S.: On the Recognition of Printed Characters of Any Font and Size. IEEE Transactions on Pattern Analysis and Machine Intelligence **9** (1987) 274–288
8. Henry S.Baird: Anatomy of a Versatile Page Reader. Proceedings of the IEEE, Special Issue on Optical Character Recognition(OCR), **80** (1992) 1059–1065
9. Bokser, M.: Omnidocument Technoloigies. Proceedings of the IEEE, Special Issue on Optical Character Recognition(OCR), **80** (1992) 1066–1078
10. Ittner, D.J., Baird, H.S.: Language-Free Layout Analysis. In: Proceedings of the 2nd ICDAR. (1993) 336–340
11. Hartley, R.T., Crumpton, K.: Quality of OCR for Degraded Text Images. In: Proceedings of the fourth ACM conference on Digital libraries. (1999) 228–229
12. Gonzalez, R.C., Woods, R.E.: Digital Image Processing. Prentice-Hall (2002)
13. Zhu, X., Yin, X.: A New Textual/Non-textual Classifier for Document Skew Correction. Proc. 16th Int'l. Conf. Pattern Recognition (ICPR'02) **1** (2002) 480–482
14. K S Sesh Kumar, Anoop Namboodiri, C V Jawahar: Learning to Segment Document Images. In: International Conference on Pattern Recognition and Machine Intelligence (to appear). (2005)
15. C. V. Jawahar, M N S S K Pavan Kumar, S S Ravi kiran: A Bilinguagl OCR for Hindi and Telugu Documents and Its Applications. In: International Conference on Document Analysis and Recognition (ICDAR). (2003) 408–412
16. Million Meshesha, C. V. Jawahar: Recognition of Printed Amharic Documents. In: International Conference on Document Analysis and Recognition (ICDAR). (2005) 784–788
17. Tapas Kanungo, Robert M. Haralick, Henry S. Baird, Werner Stuezle, David Madigan: A Statistical, Nonparametric Methodology for Document Degradation Model Validation. IEEE Transaction on Pattern Analysis and Machine Intelligence **22** (2000) 1209–1223