Journal of Computer and Communications, 2015, 3, 100-110
Published Online December 2015 in SciRes. http://www.scirp.org/journal/jcc
http://dx.doi.org/10.4236/jcc.2015.312009

Automatic Table Recognition and Extraction from Heterogeneous Documents

Florence Folake Babatunde[1], Bolanle Adefowoke Ojokoh[2*], Samuel Adebayo Oluwadare[2]

[1]Department of Computer Science, Osun State College of Education, Ila-Orangun, Nigeria
[2]Department of Computer Science, Federal University of Technology, Akure, Nigeria
Email: *bolanleojokoh@yahoo.com

Received 17 November 2015; accepted 27 December 2015; published 30 December 2015

Abstract

This paper examines automatic recognition and extraction of tables from a large collection of heterogeneous documents. The heterogeneous documents are initially pre-processed and converted to HTML codes, after which an algorithm recognises the table portion of the documents. Hidden Markov Model (HMM) is then applied to the HTML code in order to extract the tables. The model was trained and tested with five hundred and twenty six self-generated tables (three hundred and twenty-one (321) tables for training and two hundred and five (205) tables for testing). Viterbi algorithm was implemented for the testing part. The system was evaluated in terms of accuracy, precision, recall and f-measure. The overall evaluation results show 88.8% accuracy, 96.8% precision, 91.7% recall and 88.8% F-measure revealing that the method is good at solving the problem of table extraction.

Keywords

Hidden Markov Model, Table Recognition and Extraction, Hypertext Markup Language, Heterogeneous Documents

1. Introduction

A table is made up of series of rows and columns. It is a cell that houses information on a table, the information can be alphabetic, numeric or alphanumeric; a table can also house images depending on what the user wants to use the table for. In sciences, tables are used to summarize a topic or a statement, also, in accounts; tables are used to know the money spent, the money at hand and the money to be spent. So, a table can be referred to as a set of facts or figures arranged in lines or columns.

*Corresponding author.

A table as a list of facts or numbers arranged in a special order, usually in rows and columns and as an arrangement of numbers, words or items of any kind, in a definite and compact form, so as to exhibit some set of facts or relations in a distinct and comprehensive way, for convenience of study, reference, or calculation.

According to [1], tables are visual oriented arrangements of information widely used in many different domains as a way to present and communicate complex information to human readers. Tables are used in all facets of lifelike science, engineering and agriculture among others. Tables appear in print media, handwritten notes, computer software, architectural ornamentation, traffic signs and many other places and have been described as databases designed for human eyes [2]. Tables appear in the earliest writing on clay tablets, and in the most modern Web pages. Some make use of line-art, others rely on whitespace only. The precise conventions and terminology for describing tables varies depending on the context. Further, tables differ significantly in variety, structure, flexibility, notation, representation and use. However, tables have many different layouts and are mostly contained in semi-structured and unstructured documents having various internal encodings such as HTML, PDF and flat text. Table recognition is concerned about detection of all lines, both vertical and horizontal which make up a table, along with their intersection, which will aid not only to detect a table which consequently can be extracted out of a whole document but also to describe both the physical and logical structure, thus, inferring a table recognition process [3]. While it is easy for human beings to recognise and understand tables, it is difficult for computers, because tables do not have any identifying characteristic in common (because tables do not have the same identifying characteristic) [4].

Table Extraction (TE) is the task of detecting and decomposing table information in a document [5]. Automatically extracting information contained in tables and storing them in structured machine-readable (usable) form is of paramount importance in many application fields. Table recognition and extraction is a very challenging problem that poses many issues to researchers and practitioners in defining effective approaches.

Among many inspirational works is the work of [6] who designed an automatic table metadata extraction algorithm which was tested on PDF and that of [7] that extracted metadata from heterogeneous references using hidden Markov model and performed smoothing for transition probabilities.

2. Related Works

Many researchers have used HMM to extract one thing or the other from documents. Ojokoh *et al*. [7] used a full second-order HMM to extract metadata from heterogeneous references. Ana Costa e Silva [8] experimented with different ways of combining the components of an HMM for document analysis applications, in particular for finding tables in text. Pinto *et al*. (2003) in their own work presented the use of conditional random fields (CRFs) for table extraction, and compared them with hidden Markov models (HMMs). One of the pioneering works on table detection and recognition was done by [9]. Kieninger *et al*. [9] developed a table spotting and structure extraction system called T-Recs. The system relies on word bounding boxes as input. These word boxes are clustered with a bottom-up approach into regions by building a "segmentation graph". These regions are then designated as candidate table regions if they satisfy certain criterion. The key limitation of the approach is that, based only on word boxes, multi-column layouts cannot be handled very accurately, so, it works well for only single column pages.

Oro *et al*. [1] automatically extracted information contained in tables and stored them in structured machine-readable form. Dalvi *et al*. [10] described an open-domain information extraction method for extracting concept-instance pairs from an HTML corpus. The method used relies on a novel approach for clustering terms-found in HTML tables, and then assigning concept names to these clusters using Hearst patterns. The method was efficiently applied to a large corpus, and experimental results on several datasets showed that the method can accurately extract large numbers of concept-instance pairs, the problem of automatically harvesting concept-instance pairs from a large corpus of HTML tables was considered.

Tengli *et al*. [11] extracted table automatically from web pages of semi-structured HTML tables, learns lexical variants from various examples but uses vector space model to deal with non-exact matches and labels. They defined their own evaluation methodology because they did not encounter any experimental evaluation of table extraction that can be used as reference of comparison with their work. Sale *et al*. [12] focused on extracting tables from large-scale HTML texts, analysed the structural aspects of a web table within which they devised the rules to process and extract attribute-value pairs from the table. Heuristic rules were employed to identify the tables. Also, a table interpretation algorithm was proposed which captures the attribute-value relationship among table cells. Yildiz *et al*. [4] developed several heuristics which together recognise and decompose tables in PDF files and store the extracted data in a structured data format (XML) for easier reuse. A prototype was also implemented which gives the user the ability of making adjustments on the extracted data. The work shows that purely heuristic-based approaches can achieve good results, especially for lucid tables. Wang and Hu [13] considered the problem of table detection in web documents and described a machine learning based approach to classify each given table entity as either genuine or non-genuine. Gatos *et al*. [3] proposed a novel technique for automatic table detection in document images, they did not use either training phase or domain-specific heuristics but used an approach which applied to a variety of document types, that is, used a particular parameterization that depends on the average character height of the document image. Their proposed approach builds upon several consequent stages such as image preprocessing; horizontal and vertical line detection (horizontal and vertical line estimation and (ii) line estimation improvement by using image/text areas removal.); and table detection (Detection of line intersections and (ii) table detection-reconstruction.). The efficiency of the proposed method is demonstrated by using a performance evaluation scheme which considers a great variety of documents such as forms, newspapers/magazines, scientific journals, tickets/bank cheques, certificates and handwritten documents.

HMM is one of the most popular models being used for sequential model, it is widely used because it is simple and easy enough that one can actually estimate the pre-eminence, it is also rich enough that it can handle real world applications. HMM is a doubly stochastic process with an underlying stochastic process that is not observable (it is hidden), but can only be observed through another set of stochastic processes that produce the sequence of observed symbols [14].

Information can be extracted using standard approaches of hand-written regular expressions (perhaps stacked), using classifiers (like generative: naïve Bayes classifier, discriminative: maximum entropy models), sequence models (like Hidden Markov model, Conditional Markov model (CMM)/Maximum-entropy Markov model (MEMM), Conditional random fields (CRF) (are commonly used in conjunction with IE for tasks as varied as extracting information from research papers to extracting navigation instructions). Hybrid approaches can also be used for IE which is the combination of some of the standard approaches listed above but HMM was used in this research work.

This research work was therefore motivated by the need to recognize and extract tables from a large collection of documents of different types and structures and it was able to achieve its aim upon converting the different faces of the documents to similar faces using document converter.

It has been shown in this work that HMM is a fast learner even with small size data set and can also do very well with a large size data set since it is a machine learning model.

3. Hidden Markov Model

A Hidden Markov Model (HMM) is a finite state automation comprising of stochastic state transitions and symbol emissions. The automation models a probabilistic generative process whereby a sequence of symbols is produced by starting at a designated start state, transitioning to a new state, emitting one of a set of output symbols selected by that state, transitioning again, emitting another symbol, and so on, until a designated final state is reached. Associated with each of a set of states is a probability distribution over the symbols in the emission symbol and a probability distribution over its set of outgoing transitions [7].

Expressing the model mathematically, we have the definitions in Equations (1) to (6)

The Hidden Markov Model is a five-tuple $(S, A, \prod, V, B)$,

$$H = (S, A, \prod, V, B) \tag{1}$$

S is the set of states,

$$S = \{S_1, \cdots, S_N\} \tag{2}$$

$$A = \{a_{i,j}\} \text{ is the probability of being from state } i \text{ to state } j \tag{3}$$

where,

$$\sum_{j=1}^{N} a_{i,j} = 1,\ 1 \le i \le N$$

$$\Pi = \{P[S_1(1)]\}, \text{ indicating the probability of being at state } S_i \text{ at time } t = 1 \tag{4}$$

$$V = \{V_1, \cdots, V_m\} \tag{5}$$

where M is the number of emission symbols in the discrete vocabulary, V.

$$B = \{b_j(V_k)\}, \text{ indicating the probability of observing symbol, } V_k \text{ at state } S_j, \tag{6}$$

where,

$$\sum_{k=1}^{M} b_j(V_k) = 1, \ 1 \le j \le N$$

A table contains properties like td, tr, tbody, table, paragraph (that is table data, table row, table body, table, paragraph respectively). Each of these properties is a state generating sequence of observable/emission symbols during transition from one state to another. Given an HMM, each transition is performed by determining the sequence of states that is most likely to have generated the entire observable/emission symbol sequence. Viterbi algorithm is a common way of showing the most probable path of (for) the sequence [7]. After the Viterbi algorithm had shown the most probable path, the table extracted by the HMM was shown while its HTML version was generated with the help of an algorithm shown below:

1) For all the tables generated, pick a table ($t_i \quad \forall i = 0,1,\cdots,n-1$. where n = number of tables) if they are not exhausted.
2) Display the fetched table (t_i)
3) Generate the HTML (h_i) for t_i
4) For all the tags ($k_j \quad \forall j = 0,1,\cdots,m-1$. where m = number of tags in tables t_i); fetch a tag.
5) Display the tag k_j
6) Fetch the next token.
    a) If it is not a tag but inner HTML, indent positively and display content.
    b) If it is another tag indent positively (indent right) and display content.
    c) If it is closing tag indent negatively and display content.

The tables extracted by the HMM were sometimes not the same as the original tables but may be different in either of the following:

1) the contents of two cells may merge into a single cell (that is, the content of cell B may be extracted into cell A leaving cell B empty and adding to the content of cell A and vice versa).
2) a cell may be missing,
3) a row may be missing as the case may be.

But in all, there is retention of about 90% to 98% of the original table.

4. System Architecture

The system architecture shown in **Figure 1** describes the table recognition and extraction processes. The heterogeneous documents constitute the source document. These documents were pre-processed where necessary and parsed in order to extract tables that were embedded in them for further processing.

The design spans through all the tools used for pre-processing heterogeneous documents into HTML code, the generation of Transition Matrix, Initial Probability Matrix, Observable Symbol Matrix, Smoothing the Observable Symbol Matrix using Laplace Smoothing Method and Viterbi's algorithm to determine the best path, and finally the extracted tables and contents.

4.1. The Training Phase

The data sets used in this research work were self-generated because there is no known standardised data set for table extraction. It contains five hundred and twenty six (526) tables in twenty-five (25) documents of Microsoft Word, Portable Document Format and Hypertext Markup Language. This set of documents is divided into two phases: The training phase and the testing phase. In document pre-processor, heterogeneous documents were converted to their HTML equivalence. The PDF is first converted to Word document using a PDF converter (PDFC) before being converted to its HTML equivalence.

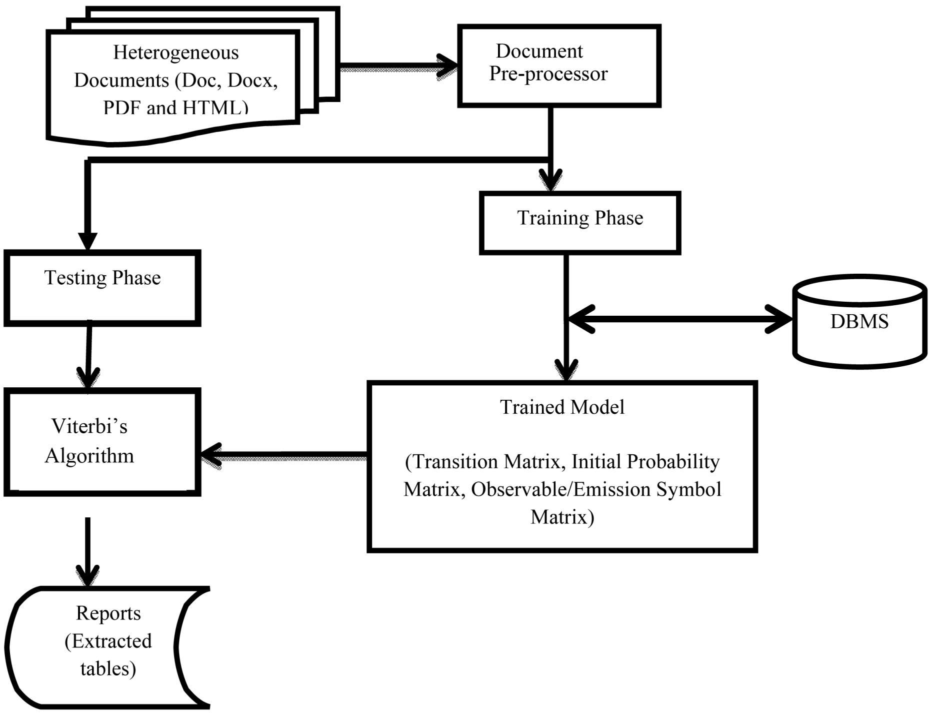
Heterogeneous Documents (Doc, Docx, PDF and HTML)
Document Pre-processor
Training Phase
Testing Phase
DBMS
Viterbi's Algorithm
Trained Model
(Transition Matrix, Initial Probability Matrix, Observable/Emission Symbol Matrix)
Reports (Extracted tables)

Figure 1. Architecture for table recognition and extraction processes.

To build an HMM for table extraction in documents, one needs to first decide how many states the model will contain and what transitions between states should be allowed [7]. At the beginning of the training phase, the model was automatically learned from the training data. The model was first constructed to produce all the results for Transition Probability Matrix, Initial Probability Matrix and Observable/Emission Symbols/Words contained in the data sets, with the start state having a transition being represented by a unique path with one state per tag. Three hundred and twenty-one (321) tables were used. As the training data are coming from the training phase, they may be saved in the DBMS or may not be saved. This model consists of the result for Transition Probability Matrix, Initial Probability Matrix, Observable/Emission Symbol Matrix and the smoothened Observable/Emission Symbol Matrix to avoid a negative result at the end. Transition Probability Matrix like the name implies is a matrix of state transitions. It has the general mathematical representation as demonstrated.

Given:

$$S = \begin{pmatrix} s_{11} & s_{12} & s_{13} & s_{14} & \cdots & s_{1n} \\ s_{21} & s_{22} & s_{23} & s_{24} & \cdots & s_{2n} \\ s_{31} & s_{32} & s_{33} & s_{34} & \cdots & s_{3n} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ s_{m1} & s_{m2} & s_{m3} & s_{m4} & \cdots & s_{mn} \end{pmatrix} \quad \text{where } S \text{ is a matrix of states}$$

$$T = \begin{pmatrix} t_{11} & t_{12} & t_{13} & t_{14} & \cdots & t_{1n} \\ t_{21} & t_{22} & t_{23} & t_{24} & \cdots & t_{2n} \\ t_{31} & t_{32} & t_{33} & t_{34} & \cdots & t_{3n} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ t_{m1} & t_{m2} & t_{m3} & t_{m4} & \cdots & t_{mn} \end{pmatrix} \quad \text{where } T \text{ is a matrix of state transitions}$$

This is a system of n states.

For all $t_{i,j}$ which is the probability of each state transition, the sum along all the rows is 1, and the transitions are non-negative. A transition probability that is zero (0) signifies an impossible transition from any state to that state. All probabilities cannot be greater than unity.

The case study has five states: TD, TH, TBody, P, Table while the algorithm for the transition probability is given below:

1) Get the object document from the CK Editor using get Data() API function
2) Determine which of the states of interest comes first
3) Find the probability of each of the states considering the number of documents used
4) Store this in a table.
5) Output the initial probability matrix.

Observable/emission symbols are the symbols extracted by the system.

In this research work, the following are the categories of observable symbols:

1) Numbers
2) Single words
3) Short statements
4) Long statements
5) Title case statements
6) Uppercase statements

The probability of the observable/emission symbols was computed and the algorithm computes the probability of the observable/emission symbols, stores its matrix and generates an output.

A description of the observable/emission symbols and samples from the data set is shown in **Table 1**.

4.2. Smoothing

Since the observable/emission symbol probability assigns zero probability to some unseen emissions, retaining these zeroes would affect the result negatively, and then there is a need for smoothing. There are so many methods used for smoothing, but in this research work, Laplace method was used. The maximum likelihood model was modified such that one (1) is placed as the numerator and m is added to the denominator [15].

$$b_{i,j}(V_k) = \frac{1}{\sum \gamma(j) + m} \tag{7}$$

where V_k is the unseen symbol while transitioning from state i to j, m is the addition of the total number of sequence of states and the number of non-zero probability across row.

4.3. The Testing Phase

This phase contains two hundred and five (205) tables. This is used for the evaluation of the trained model.

4.3.1. The Viterbi Algorithm

After going through the processes of transitioning, transition probability, observable/emission probabilities, a trained Hidden Markov Model would have been formed. This model consisting of values for the transition and observable/emission probability matrices alongside with the sequence of symbols derived from the testing data sets will be passed to the Viterbi Algorithm which will produce a sequence of states ($q_1, \cdots, q_n$) most likely to have produced the symbol sequence which represent the tokens in the table.

Table 1. Observable/emission symbols and examples.

S/N	Emission Symbols	Examples from sample
1	Numbers	2, 3, 4
2	Words	Programming, Work done, Thanks
3	Short statements	ID card automation, Database design
4	Long statements	I have personally embarked on some projects and also embarked on some other projects as a member of a team. The following are some of the projects
5	Title case statements	Number of Staff in Statistics
6	Capital letter statements	FEMI VINCENT,REMI OLADAPO

The specific problem to be solved by Viterbi algorithm is to obtain the most possible state sequence, $Q=\{q_1,q_2,\cdots,q_T\}$ for a given observation sequence, $O=\{O_1,O_2,\cdots,O_T\}$. This can be obtained by computing the probabilities for all possible sequences and then obtaining the maximum probability as follows:

$$\delta_t(i)=\max P\left[q_1,q_2,\cdots,q_t=I,O_1,O_2,\cdots,O_t \,|\, \lambda\right] \quad (8)$$

The Viterbi algorithm makes a number of assumptions. First, both the observed events and hidden events must be in a sequence. This sequence often corresponds to time. Second, these two sequence need to be aligned and an instance of an observed event needs to correspond to exactly one instance of a hidden event. Third, computing the most likely hidden sequence up to a certain point *t* must depend only on the observed event at point *t*, and the most likely sequence at point *t*-1. These assumptions listed above can be elaborated as follows. The Viterbi algorithm operates on a state machine assumption. That is, at any time the system being modeled is in some state. There are a finite number of states, however large, that can be listed. Each state is represented as a node. Multiple sequences of state (paths) can lead to a given state, but one is the most likely path to that state called the "survivor path." This is a fundamental assumption of the algorithm because the algorithm will examine all possible paths leading to a state and only keep the one most likely. This way the algorithm does not have to keep track of all possible paths, only one per state. A second key assumption is that a transition from a previous state to a new state is marked by an incremental metric, usually a number. This transition is computed from the event. The third key assumption is that the events are cumulative over a path in some sense, usually additive. So the crux of the algorithm is to keep a number for each state. When an event occurs, the algorithm examines moving forward to a new set of states by combining the metric of a possible previous state with the incremental metric of the transition due to the event and chooses the best. The incremental metric associated with an event depends on the transition possibility from the old state to the new state.

The Viterbi algorithm is stated as follows in four major steps:

1) Initialization: $\delta_1(i)=\prod_i b_i(O_1),\ 1\le i\le N$ (9)

$$\Psi_1(i)=0 \quad (10)$$

2) Recursion: $\delta_t(j)=\max_{1\le i\le N}\left[\delta_{t-1}(i)a_{i,j}\right]b_j(O_t)$ (11)

$$\Psi_t(j)=\arg\max_{1\le i\le N}\left[\delta_{t-1}(i)a_{i,j}\right] \quad (12)$$

$$2\le t\le T,\ 1\le j\le N$$

3) Termination: $P^*=\max_{1\le i\le N}\left[\delta_T(i)\right]$ (13)

$$q_T^*=\arg\max_{1\le i\le N}\left[\delta_T(i)\right] \quad (14)$$

4) Path (state sequence) backtracking

$$q_t^*=\Psi_{t+1}\left(q_{t+1}^*\right),\ t=T-1,T-2,\cdots,1 \quad (15)$$

4.3.2. Reports (Extracted Tables)

The result of the best path given by the Viterbi's algorithm is interpreted to its equivalent table in HTML code for proper evaluation.

4.3.3. Evaluation

The testing data set was evaluated to see how well the trained model performed the task of table extraction.

Evaluation was done measuring per token accuracy, precision, recall and f-measure for each extracted table in the tested documents. These evaluation measures are defined in Equations (16)-(21).

$$\text{Precision}=\frac{A}{A+C} \quad (16)$$

$$\text{Recall}=\frac{A}{A+B} \quad (17)$$

$$\text{Accuracy} = \frac{A + D}{A + B + C + D} \tag{18}$$

$$F\text{-measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{19}$$

$$\text{Overall Precision} = \frac{\sum_{i=1}^{n} A_i}{\sum_{i=1}^{n} A_i + C_i} \tag{20}$$

$$\text{Average Precision} = \frac{\sum_{i=1}^{n} \text{Precision}_i}{n} \tag{21}$$

where A is number of correctly extracted cells (True Positives), B is number of cells existing but not extracted (False Negatives), C is number of cells extracted but associated with wrong labels (False Positives) and D is number of cells that did not exist and were not extracted (True Negatives).

4.3.4. Experimental Results and Discussion

Experiments were carried out using a self-generated data set. The training set was used to train the HMM and the testing set was used to evaluate the effect of extraction.

Table 2 shows the evaluation results of the hidden Markov model for automatic table extraction from heterogeneous documents from the dataset. Document 16 had the highest accuracy (93.0%), precision (98.2%), recall (94.3%) and f-measure (95.8%) values because it had the highest number of tables. Document 21 had higher values for accuracy (92.3%), precision (98.1%), recall (93.8%) and f-measure (95.6%) which had 21 tables than document 20 which had 23 tables with values for accuracy (85.0%), precision (94.5%), recall (88.2%) and f-measure (85.9%). The reason is because document 21though having 21 tables had some tables that are lengthy while document 20 does not have any lengthy table among the 23 tables it had. Documents 24 and 25 had the same number of tables and the least but document 25 had better values for accuracy (81.5%), precision (91.8%) and f-measure (87.4%) than document 24 while the two documents had the same recall value (85.2%). The table also showed that HMM is a fast learner. It had learned effectively for as low as 66 tables (with higher accuracy, precision, recall and f-measure than for one hundred and thirty-nine (139) tables and two hundred and five (205) tables). Increase in the number of tables seems not to be proportional to the effectiveness of the learning process of the model, as the overall and average accuracy, precision, recall and f-measure are higher for one hundred and thirty-nine (139) tables than for two hundred and five (205) tables.

Table 3 shows the comparison between Trigram HMM and Table Extraction from Heterogeneous Documents

Table 2. Accuracy, precision, recall and F-measure of HMM from self-generated data set from each document.

Document	Accuracy	Precision	Recall	F-measure
16	0.930	0.982	0.943	0.958
17	0.877	0.965	0.898	0.895
18	0.816	0.938	0.856	0.850
19	0.680	0.933	0.865	0.783
20	0.850	0.945	0.882	0.859
21	0.923	0.981	0.938	0.956
22	0.816	0.926	0.847	0.844
23	0.836	0.931	0.870	0.876
24	0.805	0.911	0.852	0.871
25	0.815	0.918	0.852	0.874
Overall for 526 tables	0.888	0.968	0.917	0.888
Average for 526 tables	0.821	0.935	0.842	0.878
Overall for 66 tables	0.922	0.980	0.937	0.935
Average for 66 tables	0.885	0.966	0.903	0.919
Overall for 139 tables	0.852	0.955	0.897	0.864
Average for 139 tables	0.790	0.920	0.814	0.859

(Table Extractor) using HMM. It shows that the precision for Table Extractor HMM (the proposed system) is higher than that of Trigram HMM [7] with 3.3%. Recall and F-measure were better with Trigram HMM than that of Table Extractor HMM. The reason being that different experiments were performed by different authors with different datasets also, the number of data in the dataset used by each author differs.

Figure 2 shows that the proposed system has a good accuracy and recall, but better precision. **Figure 3** shows that the proposed system has 93.5% average precision, 82.1% accuracy with a recall of 84.2%. **Figure 4** shows comparison of Overall Precision, Accuracy, Recall and F-measure for different number of tables used for the experiment, that is, two hundred and five tables which is the total number of tables used for testing which is divided into two: sixty six and one hundred and thirty nine tables. Sixty six (66) tables have the highest accuracy, precision, recall and f-measure of 92.2%, 98%, 93.7% and 93.5% respectively.

Figure 5 shows comparison of Average Accuracy, Precision, Recall and F-measure for different number of tables used for the experiment, that is, two hundred and five tables which is the total number of tables used for testing which is divided into two: sixty-six and one hundred and thirty-nine tables. 66 tables have the highest accuracy, precision, recall and f-measure of 88.5%, 96.6%, 90.3% and 91.9% respectively.

Table 3. Comparison of precision, recall and f-measure of the trigram HMM and table extractor HMM.

	Precision	Recall	F-measure
Trigram HMM	93.5	93.2	93.4
Table Extractor HMM	96.8	91.7	88.8

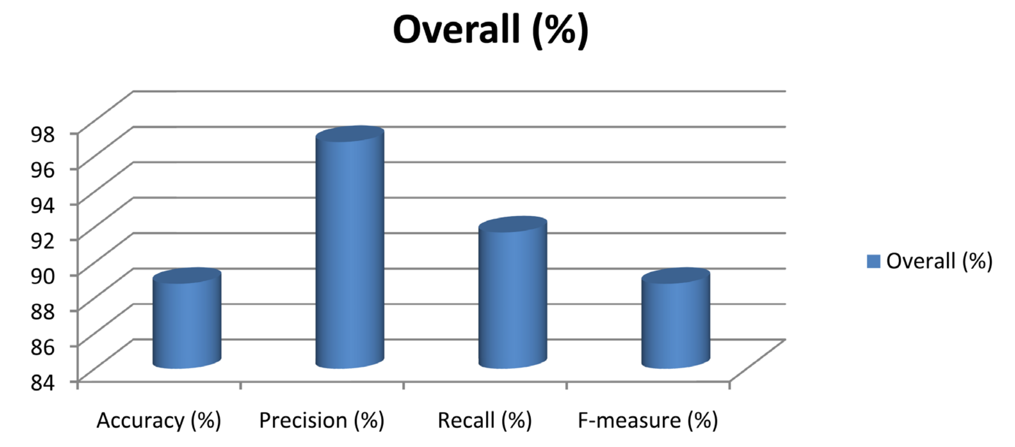

Figure 2. Overall accuracy, precision, recall and f-measure in percentage.

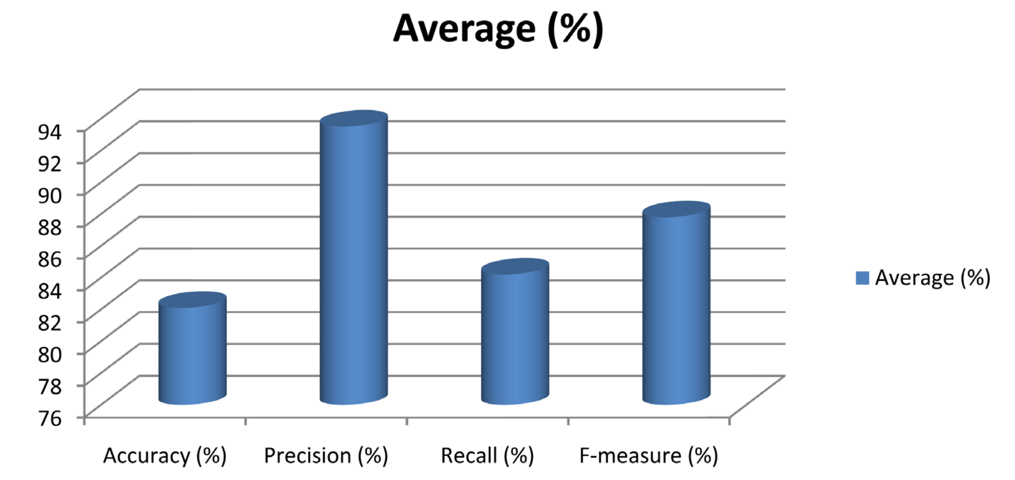

Figure 3. Overall average for accuracy, precision, recall and f-measure in percentage.

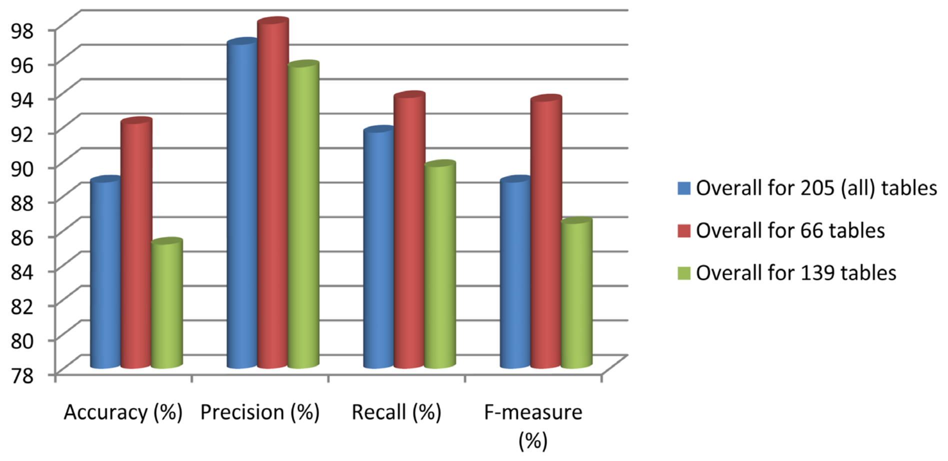

Figure 4. Comparison of overall precision, accuracy, recall and F-measure for different number of tables.

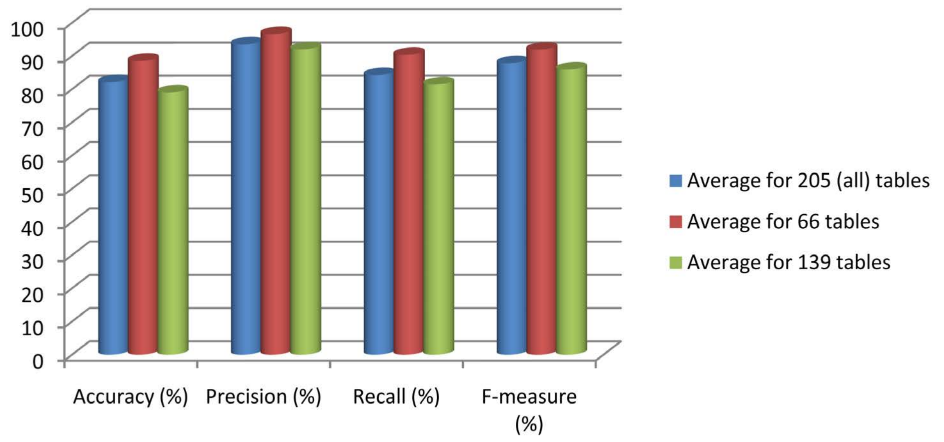

Figure 5. Comparison of average precision, accuracy, recall and f-measure for different number of tables.

5. Conclusion and Recommendations

This research work presented how tables can be recognised and extracted automatically from heterogeneous documents using Hidden Markov Model (HMM). Smoothing was done for transition probability matrix and Viterbi's algorithm was used to get the best path with the use of an algorithm which helped to display both the extracted table and its HTML equivalence; these were stored in a database and also displayed in web browsers. Automatic table recognition and extraction provide scalability and usability for digital libraries and their collections. Heterogeneous documents (except HTML documents) were initially pre-processed and converted to HTML codes after which an algorithm recognises the table. HMM was applied to extract the table portion from HTML code. The model was trained and tested with five hundred and twenty-six self-generated tables (three hundred and twenty-one (321) tables for training and two hundred and five (205) tables for testing). Viterbi algorithm was implemented for the testing. Evaluation was done to determine the accuracy, precision, recall and f-measure of the extraction.

In this research work, only Word, PDF and HTML documents were used; future work could accommodate other types of documents like Excel, PowerPoint and so on. Modification can also be done to this work in the nearest future to see that not only Google Chrome is able to do the extraction completely, but all internet browsers. A four-level cross validation is suggested for future work. Other means of validation apart from accuracy, precision, recall and f-measure can be used for evaluation.

References

[1] Oro, E. and Ruffolo, M. (2009) PDF-TREX: An Approach for Recognizing and Extracting Tables from PDF Documents. 10*th International Conference on Document Analysis and Recognition*, Barcelona, 26-29 July 2009, 906 - 910. http://dx.doi.org/10.1109/icdar.2009.12

[2] Pinto, D., McCallum, A., Wei, X. and Bruce, W. (2003) Table Extraction using Conditional Random Fields. *Proceedings of the* 26*th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, New York, 235-242. http://dx.doi.org/10.1145/860435.860479

[3] Gatos, B., Danatsas, D., Pratikakis, I. and Perantonis, S.J. (2005) Automatic Table Detection in Document Images. *Proceedings of the International Conference on Advances in Pattern Recognition*, Bath, 22-25 August 2005, 612-621.

[4] Yildiz, B., Kaiser, K. and Miksch, S. (2005) pdf2table: A Method to Extract Table Information from PDF Files. *Proceedings of the* 2*nd Indian International Conference on Artificial Intelligence*, Pune, 20-22 December 2005, 1773-1785. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.94.9382

[5] Cafarella, M.J. (2009) Extracting and Managing Structured Web Data. Ph.D. Dissertation on Computer Science and Engineering, University of Washington, Seattle.

[6] Liu, Y., Mitra, P. and Giles, L.C. (2006) TableSeer: Automatic Table Metadata Extraction and Searching in Digital Libraries. *Proceedings of the* 6*th ACM/IEEE-CS Joint Conference on Digital Libraries*, ACM, New York, 339-340. http://dx.doi.org/10.1145/1141753.1141835

[7] Ojokoh, B., Zhang, M. and Tang, J. (2011) A Trigram Hidden Markov Model for Metadata Extraction from Heterogeneous References. *Journal of Information Sciences*, **181**, 1538-1551. http://dx.doi.org/10.1016/j.ins.2011.01.014

[8] Costa e Silva, A. (2009) Learning Rich Hidden Markov Models in Document Analysis: Table Location. *Proceedings of the International Conference on Document Analysis and Recognition*, Barcelona, 26-29 July 2009, 843-847.

[9] Kieninger, T.G. (1998) Table Structure Recognition Based on Robust Block Segmentation. *Proceedings of SPIE* 3305, Document Recognition V, 22-32.

[10] Dalvi, B., William, W., Cohen, J. and Callan, J. (2012) WebSets: Extracting Sets of Entities from the Web Using Unsupervised Information Extraction. *Proceedings of the* 5*th ACM International Conference on Web Search and Data Mining* (*WSDM*), Seattle, 8-12 February 2012, 245-252.

[11] Tengli, A., Yang, Y. and Ma, N.L. (2004) Learning Table Extraction from Examples. *Proceedings of the* 20*th International Conference on Computational Linguistics*, Association for Computational Linguistics, Stroudsburg, Article No. 987. http://dx.doi.org/10.3115/1220355.1220497

[12] Sale, M.A., Chawan, P.M. and Chauhan, P.M. (2012) Information Extraction from Web Tables. *International Journal of Engineering Research and Applications* (*IJERA*), **2**, 313-318.

[13] Wang, Y. and Hu, J. (2002) A Machine Learning Based Approach for Table Detection on the Web. *Proceedings of the* 11*th International Conference on World Wide Web* (*WWW*), ACM, New York, 242-250. http://dx.doi.org/10.1145/511446.511478

[14] Rabiner, L.R. and Juang, B.H. (1986) An Introduction to Hidden Markov Models. *IEEE ASSP Magazine*, **3**, 4-16. http://dx.doi.org/10.1109/MASSP.1986.1165342

[15] Borkar, V.R., Deshmukh, K. and Sarawagi, S. (2001) Automatic Segmentation of Text into Structured Records. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, ACM, New York, 175-186. http://dx.doi.org/10.1145/375663.375682