A Hybrid Intelligence Approach to Artifact Recognition in Digital Publishing

J. Fernando Vega-Riveros, fvega@ece.uprm.edu
Hector J. Santos Villalobos, hector.santos@ece.uprm.edu
Department of Electrical and Computer Engineering
University of Puerto Rico, Mayagüez

Abstract – The system presented integrates rule-based and case-based reasoning for artifact recognition in Digital Publishing. In Variable Data Printing (VDP) human proofing could result prohibitive since a job could contain millions of different instances that may contain two types of artifacts: 1) evident defects, like a text overflow or overlapping 2) style-dependent artifacts, subtle defects that show as inconsistencies with regard to the original job design. We designed a Knowledge-Based Artifact Recognition tool for document segmentation, layout understanding, artifact detection, and document design quality assessment. Document evaluation is constrained by reference to one instance of the VDP job proofed by a human expert against the remaining instances. Fundamental rules of document design are used in the rule-based component for document segmentation and layout understanding. Ambiguities in the design principles not covered by the rule-based system are analyzed by case-based reasoning, using the Nearest Neighbor Algorithm, where features from previous jobs are used to detect artifacts and inconsistencies within the document layout. We used a subset of XSL-FO and assembled a set of 44 document samples. The system detected all the job layout changes, while obtaining an overall average accuracy of 84.56%, with the highest accuracy of 92.82%, for overlapping and the lowest, 66.7%, for the lack-of-white-space.

1. INTRODUCTION

Hybrid systems make use of more than one Artificial Intelligence technique and have proven to address and solve problems involving imprecision, uncertainty and vagueness, high dimensionality, and the need to reconcile both analog and symbolic computation [4], which result too complex for conventional approaches or for single AI techniques. The system described in this paper integrates rule-based and case-based reasoning techniques for artifact recognition in Digital Publishing.

Digital Publishing (DP) can be defined as a printing-imaging processes where the film or plate making stages are eliminated, and where printing or imaging takes place after the pre-press process [8]. It is an end-to-end workflow where customers can obtain the desired service on-demand. In other words DP allows publishing anything, anytime, anywhere, and by anyone, empowering individuals to control all, or most of the publishing processes [6]. A diagram of a DP workflow model is shown in Figure 1.

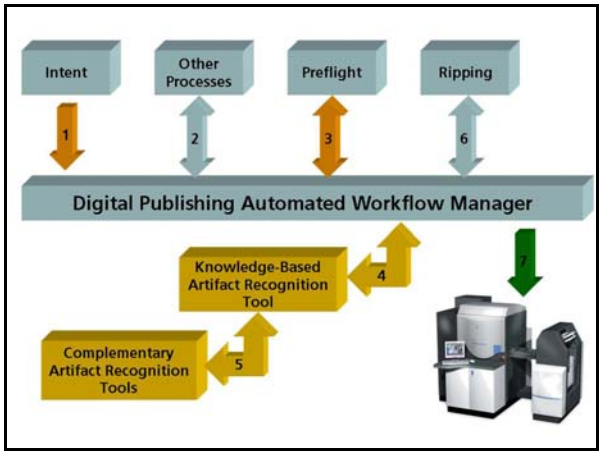

Figure 1: A Digital Publishing Workflow Model

One of the advances that DP has promoted is Variable Data Printing (VDP). The essence of VDP is job personalization where a document layout yields a template whose contents can be filled in with different images, figure illustrations, or texts for each individual reader [5]. A job of this type with multiple similar instances is known as a Variable Data Job (VDJ).

Print-shops collect basic information about the customer and the job in the Intent stage, and use two processes, preflight and proofing, before sending any job to ripping (Raster Image Processing) and to the printer. Preflight checks that the digital document contains all the elements required to perform well in the production workflow, while the proofing task is for visually inspecting a sample output for defects before printing the complete job. Proofing, as currently conducted by human experts to ensure the quality of every job, raises a problem when dealing with VDJs. If the proofing expert wants to ensure that each and every instance of a VDJ is ready for printing, he/she should visually inspect each one, a task that could result prohibitive or impossible to execute, given that a single VDJ could contain millions of different instances.

Digital Documents may contain defects produced by missing context, wrong use of metrics (for example use RGB instead of CYMK as the document color space), aesthetically unpleasant context arrangement, infringement of page constraints, image resolution below requirements, etc. We have classified defects specific to VDP into of two types. On one hand we find defects that are evident, like a text overflow in a document field. On the other hand, more subtle defects are style inconsistencies with regard to the design of a particular VDJ. We refer to them as style-dependent artifacts.

Style-dependent artifacts cannot be detected by preflight tools and require a different approach, thus, we have added an artifact recognition tool to the workflow, as shown in Figure 1. This tool is composed of two conceptual modules: The Knowledge-Based Artifact Recognition (K-BAR) and the Complementary Artifact Recognition (CAR). The K-BAR is designed for document segmentation, layout understanding, artifact detection, and document design quality assessment. The CAR is a module reserved for detecting, for example, JPEG compression artifacts such as blocking, ringing, and quantization artifacts. In this paper we will focus on the K-BAR tool. We followed three steps in our approach: studied and characterized artifacts in digital variable data documents, found the most suitable knowledge representation language for the characterized artifacts, and established the techniques and models to gather, manage, and process the knowledge used for artifact recognition in variable data jobs. Document quality assessment, as conducted by K-BAR, is constrained by the proofing of one instance of the variable data job against which the remaining instances are assessed. No assumptions or constraints are made regard to how to choose the instance of the VDJ for proofing.

Fundamental rules of document design are used in the rule-based component of K-BAR which carries out the document segmentation and layout understanding tasks. Nevertheless, ambiguities in the design principles that can not be covered by a rule-based system are analyzed by case-based reasoning, using the Nearest Neighbor Algorithm, where particular features from previous jobs are gathered and used to detect artifacts and inconsistencies within the document layout design.

This paper describes a hybrid knowledge-based system architecture capable of recognizing defects in variable data jobs. The document evaluation is constrained by the proofing of one instance of the variable data job against which the remaining instances are assessed. Next sections describe the processes behind the K-BAR framework: data input specifications, VDJs file parsing, VDJs segmentation and artificial understanding processes, the detection of document changes or anomalies, the recognition of artifacts, and the actualization of system knowledge. Finally we expose some conclusions about the system.

2. THE DIGITAL PUBLISHING AUTOMATED PREFLIGHT MODEL

We have identified three relevant processes for an automated preflight system. Two of them already exist in the traditional print shop; the Intent and the Preflight. These preflight processes are responsible for capturing, sharing, managing, and analyzing data about clients and their variable data jobs. The third process, denoted *Artifact Recognition Tool*, nowadays is executed by human experts. These processes are described below.

2.1 The intent module

The intent is the first contact between the print shop and, its customers and their print jobs. This stage is responsible for gathering information about the type of job, its description, the expected quality of the printed

job, the color system used by the designers and the number of instances, among others. Moreover this module may gather information about the client's interests, audience, expertise, organization, etc. This information can be used by an automated preflight model to execute a customized analysis of the job.

Customized analysis criteria are a fundamental part of the artifact recognition system. The analysis criteria give the level of tolerance the artifact recognition tool should have for anomalies found in a given job. Information about the client expertise and the type of job can lead to different analysis criteria. Similarly, the client's organization and audience provide information that allows evaluating jobs using the same analysis criteria for similar organizations. Different customers and types of jobs require varying degrees of quality criteria. Some may need to be extremely rigorous, for example a photography magazine, while "soccer moms' calendars" may need to be evaluated with more flexible criteria. This does not mean that the print shop will give a lower quality of service to certain customer, but that some jobs may have been designed in a way that if rigorous criteria were used, many irrelevant errors or artifacts would be detected or the jobs may become unaffordable for certain types of customers.

2.2 The preflight process

The preflight tool checks for missing job components, incorrect color systems, invalid file configurations or formats, and possible flaws given the press type and configuration. Preflight applications provide a report of their analysis. This report can be useful for artifact detection. The preflight results can pinpoint vulnerable areas inside the document, where artifacts can be found. In this way the workflow manager can send part of the job to the artifact recognition tool for further evaluation and not the entire job. This can help to reduce processing time, and have a more efficient utilization of resources. The preflight report may become a second guideline for the artifact recognition process, after the metadata generated in the Intent stage.

The press is where the ripping process takes place and is a computationally expensive process. It decodes PostScript, creates an intermediate list of objects and instructions, and finally converts graphic elements into bitmaps for rendering on an output device [9]. Many job failures in digital publishing are generated in the ripping process. For these reasons it is necessary that the preflight application verifies some basic requirements to minimize the probability of failures at the ripping stage..

2.3 The artifact recognition tool

Preflight applications continue to improve in performance, quality, and automation and are practically free of human intervention. However, the proofing stage is still executed by human experts. In the proofing stage the expert checks a printed output before the actual printing takes place [9]. At this stage of the research, we do not intend to fully automate the proofing stage; with the artifact recognition tool we desire to extend the preflight capabilities while maintaining a feasible level of human intervention in Variable Data Jobs, as was discussed in the Introduction.

We will focus in this paper on the Knowledge-Based Artifact Recognition (K-BAR). The K-BAR is designed for document segmentation, layout understanding, artifact detection, and document design quality assessment. The section below discusses the Artifact Recognition Model used in our research.

3. KNOWLEDGE-BASED ARTIFACT RECOGNITION MODEL

The main purpose of the Knowledge-Based Artifact Recognition Model, showed in Figure 2, is the detection of defects or artifacts generated by inconsistencies in the document design or style. We refer to this type of artifacts as style-dependent artifacts.

We have found three important steps to develop a successful artifact recognition system; *document segmentation*, *document understanding*, and *artifact matching*. Document segmentation is in charge of taking the job raw data and starting to group logically related components given the layout of the document. The document understanding process takes the information provided by the segmentation step and establishes logical types[1] for the components and the groups obtained in the document segmentation. The artifact matching step deals with the

[1] Some examples of logical types are chapter header, sections header, captions, text, image, and table.

characterization of the inconsistencies in the document design and matching them with previously found inconsistencies.

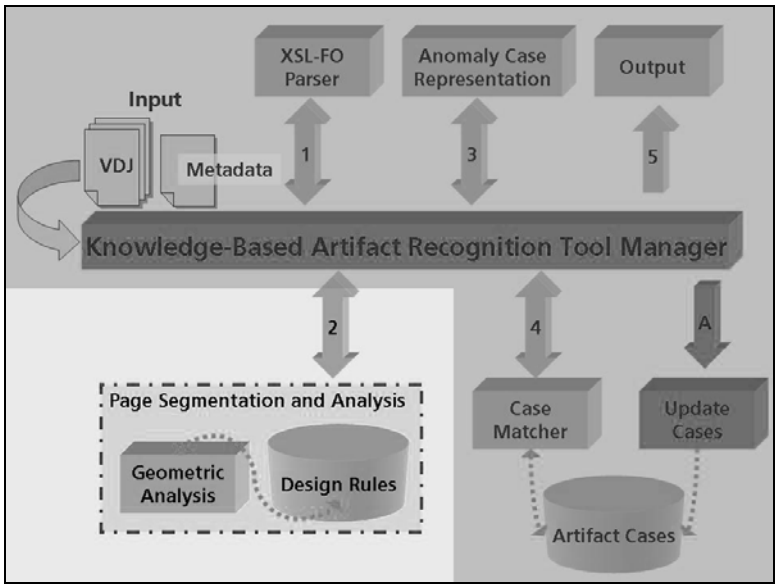

Figure 2: Knowledge-Based Artifact Recognition Model

Figure 2 shows a detailed block diagram of the Knowledge-Based Artifact Recognition Tool (K-BAR). The K-BAR tool is composed of eight principal parts: the *K-BAR Tool Manager*, the *Input*, the Job *XSL-FO* [3][1] *Parser*, the *Page Segmentation and Analysis*, the *Anomaly Case Representation*, the *Case Matcher*, the *Output*, and the *Update Cases*.

3.1 K-BAR Tool Manager

The K-BAR Tool Manager handles and supervises all the processes inside the K-BAR. This module manages the tasks needed to accomplish artifact detection and the interaction between tasks. It acts as an internal workflow engine.

3.2 Input

This module receives a variable data job and associated metadata needed for the analysis. The file format for jobs used in this research is XSL-FO (eXtensible Stylesheet Language Formatting Objects) [3][1]. This format is very versatile and can be used to format any type of digital information, such as web pages and xml data. This format contains layout information, thus simplifying the segmentation and document understanding processes.

The Variable Data Printing (VDP) job is stored in separate files, one file for each instance of the variable data job. For the proof-of-concept prototype the Input module receives the directory where the files are placed, and then the system loads all the files and starts the analysis. As explained earlier, the Artifact Recognition Tool assumes that one of the instances has been evaluated and approved by a human expert. This instance is referred to as the *approved instance*.

In addition to the directory where the VDP job files are stored, the input data contains the client type/expertise, job type and the audience the job is addressed to. This metadata is also provided in an XML file and, as explained before, plays an important role in this analysis since it is used when deciding the tolerance of the system to inconsistencies.

3.3 XSL-FO Parser

The XSL-FO Job Parser takes an XSL-FO file and extracts the geometric information of the document components, their properties, and represents this information in a data structure suitable for the system. Since XSL-FO is an XML-Based language we can search the XSL-FO nodes with Apache Xerces2's Document Object

Model[2] (DOM) API. We used DOM to represent variable data jobs because it maintains the complete document tree. To extract the properties of each node, it is required to have at any given time the properties of the node's parents, siblings and children. Properties can represent relations between document components or can be inherited from parent nodes. DOM permits us to comply with these requirements.

Our current system uses only a small subset of XSL-FO but the parser has been designed using polymorphism to make it easy to add new formatting objects. In this project we are evaluating Variable Data Jobs of one page length and any page size. Jobs can be composed by images and text. XSL-FO can be used to setup the components' size, location, alignment, and other specific properties like text typeface.

3.4 Page Segmentation and Analysis

The Page Segmentation and Analysis module takes the information extracted by the XSL-FO Parser to subdivide the document in groups of related components. It is composed of two important modules; the *Geometric Analysis* and the *Design Rules*. These two modules are co-dependent. The Geometric Analysis module extracts explicit knowledge from the components that include component position inside the page, component dimension, text size, text leading, text typeface, text color, page margins, percentage of white space on the page, and page dimension. The Geometric Analysis uses these data and data generated by the Design Rules module to flag document changes with respect to the approved instance. A detailed description of the Geometric Analysis is found in [10].

The Design Rules module is implemented as a rule-based expert system and uses the graphic design principles of repetition, proximity, alignment, similarity, and contrast [2][11][13][14][7] as the foundation for the document segmentation and understanding strategy. These principles together with the information provided by the Geometric Analysis are used to determine the relations between components which represent the designer knowledge implicit in the document layout. This allows the identification of elements such as headers, captions, footnotes, and image-caption relations. From this analysis logical units or groups of components are determined.

The components in these units are strongly related and it is assumed that the components of equivalent units have the same properties. When some change in any property is found between equivalent units the module flags the change as an *anomaly*. An anomaly is an inconsistency with regard to the document design, as found in the approved instance, or a potential artifact. For a detailed account of the Design Rule module see [10].

3.5 Anomaly Case Representation

The Anomaly Case Representation takes the analysis provided in the Page Segmentation and Analysis module to characterize the anomalies found. Each anomaly is represented as a case, where a case is a logical structure used in artificial intelligence's case-based reasoning [12]. Cases are used to characterize past events and in this research represent previously defined artifacts. Each component in each of the instances of a Variable Data Job is associated with the corresponding component in the approved instance. When anomalies are found the changes between the properties of the anomalous and the approved components are characterized. Some of the features used to characterize the artifacts are changes in page white space, component size, component dimensions, distance, component relations, and logical types.

3.6 Case Matcher

The Case Matcher module is in charge of searching through the artifacts case base to find a match for the anomaly characterized in the previous process. Cases are characterized by features whose values are determined from the difference between the properties of the potentially defective component and the equivalent object in the approved instance. We use the nearest neighbor algorithm to match cases.

We have used 21 features to represent the cases, where each feature corresponds to a axis in an 21-dimensional space. The anomaly-case is represented in the 21-dimensional space in the same manner artifact-cases are. The purpose is to measure the similarity between the anomaly-case and the artifacts in the case base by measuring the distance between the cases as given by the following equation:

[2] For more information on DOM see http://xml.apache.org/xerces2-j/dom.html

$$D(T,S) = \frac{\sum_{i=1}^{N} w_i f(T_i, S_i)}{\sum_{i=1}^{N} w_i} \qquad \text{(Equation 1)}$$

where $D(T,S)$ is the distance between the target or anomaly case T and the artifact case S, N is the number of attributes of each case, T_i is the value of the *i*-th feature of the target case, S_i is the value of the *i*-th feature of the artifact case and w_i is the value of the weight for the *i*-th attribute. The similarity function for each attribute is given by

$$f(T_i, S_i) = 1 - \frac{|T_i - S_i|}{S_i} \qquad \text{(Equation 2)}$$

For each case, the system computes the value of the similarity function. The case that is closest to the anomaly case and is below a given threshold value is considered a match. There can be times when there is no match; if this happens; we consider that the anomaly is not an artifact.

The stored cases include additional features such as a severity rating and the artifact name. These features are not used in the case matching. Other systems can use this information to create preflight reports, with the name of the artifacts and a severity rating that is not related to the artifact alone, but to the arrangement and properties of the anomalous component too. The artifact name and the severity ratings are given by a human expert at the time of adding a new case to the artifact case base. In addition this information can be used by the DP's Workflow Manager to make decisions about whether to send the job back to the client to fix errors or just send the job to the next process.

3.7 Output

The Output is another communication module. This module tells the DP Workflow manager when the artifact recognition tool has finished its analysis. This module gathers all the data generated from the detection analysis and can send it to the DP Workflow Manager, to other artifact recognition tools, or to a storage device.

3.8 Update Cases

The Update Cases module is used to add new cases to the artifact case base. This module receives an approved instance and a damaged instance. The module extracts the features of the damaged and approved components and calculates their differences. Then the human expert determines the name and severity rating of the artifact.

4. RESULTS

We designed 43 different document samples. We calculated the accuracy of the system in recognizing defects, given a set of test samples using the following equation:

$$a = \frac{E}{T} \times 100\% \qquad \text{(Equation 3)}$$

where E is the number of correct artifact detections and T is the number of artifacts that should be detected. We also estimated the probability of false alarms produced by the system, *i.e.*, wrongly detecting an artifact, using the following equation:

$$p = \frac{E}{N} \times 100\% \qquad \text{(Equation 4)}$$

where E is the number of events and N is the number of instances or components. We trained and tested the system with the following types of artifacts: typeface changes, missing components, overlapping and lack of white space. A detailed analysis of results for each type of artifact is found in [10]. For this these tests we tried 8

different weight configurations for the distance function in equation 1. The weight configurations are shown in Table 1.

Table 1 - Weight configurations

Weights Configuration								
Features	**Configuration**							
	1	*2*	*3*	*4*	*5*	6	*7*	*8*
Height	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Decrement Height	2.00	2.00	2.00	1.00	1.00	1.50	1.00	2.00
Width	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Decrement Width	2.00	2.00	2.00	1.00	1.00	1.50	1.00	2.00
Size	1.50	1.50	1.50	1.00	1.00	1.00	1.00	1.50
White Space	0.00	0.00	1.00	0.00	1.00	0.00	1.00	0.00
Decrement White Space	0.00	0.00	1.00	0.00	1.00	0.00	1.00	0.00
Above Distance	0.50	1.00	0.50	1.00	0.50	1.00	1.00	1.50
Right Distance	0.50	1.00	0.50	1.00	0.50	1.00	1.00	1.50
Below Distance	0.50	1.00	0.50	1.00	0.50	1.00	1.00	1.50
Left Distance	0.50	1.00	0.50	1.00	0.50	1.00	1.00	1.50
Margin Friendly	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Overlapping	1.00	1.00	1.00	2.00	1.50	1.50	1.00	1.00
Type	1.50	1.50	1.50	2.00	1.50	1.00	1.00	1.50
Expected Type	1.50	1.50	1.50	2.00	1.50	1.00	1.00	1.50
Type Difference	2.00	2.00	2.00	2.00	1.50	1.00	1.00	1.50
Typeface Change	2.00	2.00	2.00	2.00	1.50	1.50	1.00	2.00
Client Expertise	0	0	0	0	0	0	0	0
Audience Expertise	0	0	0	0	0	0	0	0
Audience Age Group	0	0	0	0	0	0	0	0
Audience Visual Sensitivity	0	0	0	0	0	0	0	0

These configurations were established by observation. We setup a configuration, tested the system, observed where it failed, readjusted the weights that were related to the flaw, and re-tested the system. For example, with the configuration number 7 all the features have the same significance. With this configuration, for instance, the system performed poorly with typeface change artifacts. Consequently we increased the weight for the typeface change feature. This increase led the system to a better detection of such artifacts. The configurations in Table 1 are arranged from the best performance (Configuration number 1) to the poorest performance (Configuration number 8). The personalized criteria features have weights equal to zero, because this functionality was not implemented completely in this first prototype.

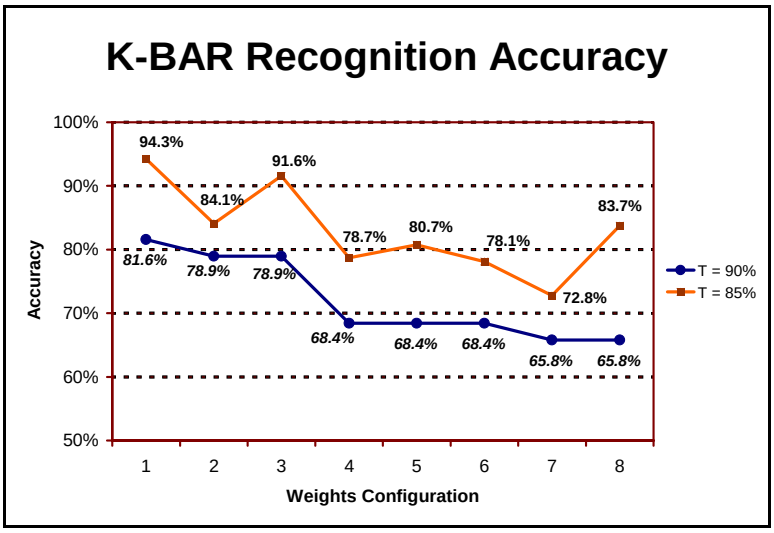

Figure 3 - Artifact recognition accuracy

Figure 3 shows the overall performance of the system in terms of accuracy for our test samples. We run the test with two thresholds, one at 85% and the other at 90%. With thresholds of 90% and 85% the system achieved accuracies of 81.8% and 94.3%, respectively. Both top performances were achieved by the same weight configuration. Notice that this configuration (Configuration number 1) has the White Space and Decrement White Space weights equal to zero since the segmentation module did not differentiate between foreground and background objects in this first implementation, which led to a reduced accuracy and increased probability of false alarms. The results with a threshold of 85% show a better performance than the results with a threshold of 90%. Nevertheless the system with a threshold of 85% generated approximately double the number of false alarms than with a threshold of 90% (See Figure 4).

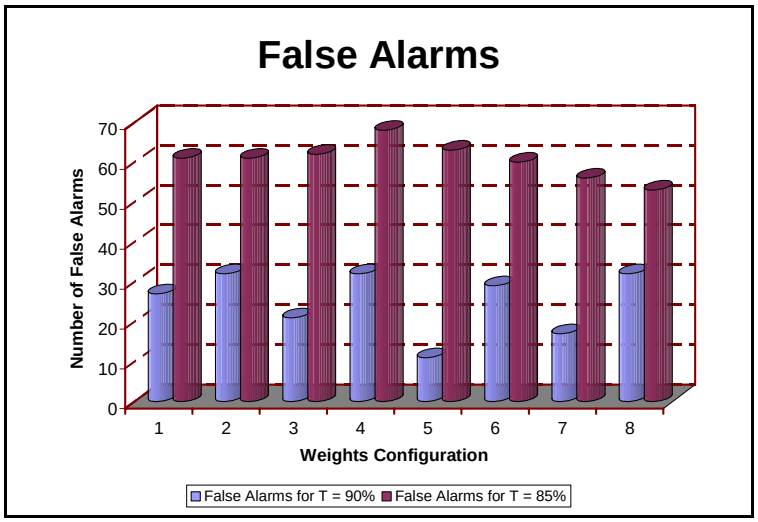

Figure 4: False alarms for different weight configurations

For the test run with a threshold of 85% the system generated a maximum of 68 false alarms, while with a threshold of 90% the false alarms decrease drastically to a maximum of 32 false alarms. These results show that the threshold plays an important role in obtaining a balance between the artifact recognition accuracy and the generation of false alarms. On one hand, lower thresholds will lead the system to a better recognition of artifacts, but more false alarms. On the other hand higher thresholds slightly decrease the accuracy of the system in detecting artifacts, but diminish drastically the generation of false alarms.

5. CONCLUSIONS

This paper presented a powerful tool for artifact recognition in Digital Publishing. The tool was designed as a hybrid knowledge-based system composed of rule-based and case-based reasoning. Rule-based techniques are used to code graphic design principles used to segment and understand the document contents, and extract the data needed to create anomaly cases. Case-based techniques are used to represent and store past knowledge about known artifacts, and detect artifacts in Variable Data Printing Jobs.

The knowledge-based artifacts recognition tool is an important technology inside the Digital Publishing workflow. Digital Publishing is a new field in the printing industry; that demands automated artifact recognition. This architecture is the first step in solving a problem which had been traditionally executed by human experts but that in Variable Data Printing may result prohibitive or impossible.

Some relevant contributions are encountered in this research. The architecture design provides for personalized analysis criteria, which gives the K-BAR a certain amount of tolerance to inconsistencies. Also, a guideline for style-dependent artifact recognition was established. Moreover this architecture provides a framework for knowledge sharing, where print shops can sell or buy artifact recognition services. Print shops can share discovered artifact cases, or can update their systems with additional design rules in order to improve the document segmentation and understanding process. The K-BAR framework is versatile, flexible, modifiable, and allows future knowledge sharing, knowledge reuse, task sharing, task distribution, and web integration.

6. ACKNOWLEDGEMENTS

This research was conducted with a grant from Hewlett Packard under the Digital Publishing Research Program in Collaboration with Purdue University and HP Labs.

7. REFERENCES

[1] Adler, S., A. Berglund, J. Caruso, S. Deach, T. Graham, P. Grosso, E. Gutentag, A. Milowski, S. Parnell, J. Richman and S. Zilles. *Extensible Stylesheet Language (XSL) Version 1.0*. W3C Recommendation 15 October 2001. http://www.w3.org/TR/2001/REC-xsl-20011015/.

[2] Collier D., “Collier’s Rules for Desktop Design and Typography”, Addison-Wesley, 1991.

[3] G. K. Holman, “Definitive XSL-FO”, Prentice Hall, 2003.

[4] Grossberg, S. Foreword. International Journal of Hybrid Intelligent Systems. Vol 1. No. 1-2. 2004. Pp 1.

[5] H. Chao and J. Fan, “Layout and Content Extraction for PDF Documents”, DAS 2004, IAPR Int. Workshop on Document Analysis Systems, Florence, Italy, 8-10 Sept. 2004.

[6] Kleper M. L., “The Handbook of Digital Publishing”, Graphic Dimensions 2001, Prentice Hall, Vol. 1, Page XXVII – XXXVI.

[7] L. Purvis, S. Harrington, B. O'Sullivan, and E. C. Freuder, “Creating Personalized Documents: An Optimization Approach”, Proceedings of the 2003 ACM symposium on Document Engineering, 2003, Grenoble France, Pages: 68 – 77, ISBN: 1-58113-724-9.

[8] Politis A., “Digital Printing: finishing technologies making digitally printed documents professional”, XML Europe 2000 Conference, 12 – 16 June 2000, Paris France.

[9] Rivera, W., M. Rodriguez-Martinez, N. Santiago, F. Vega, T. Avellanet, G. Chaparro, W. Lozano, A. Pereira, and H. Santos-Villalobos, “Towards Development of Concepts and Algorithms to Enable Automated Digital Publishing Workflows”

[10] Santos-Villalobos, H. J. *Style-dependent Artifact Recognition for Digital Variable Data Printing*. M.Sc. Thesis. University of Puerto Rico. July 2005.

[11] Topping S. M., “Graphic design and color in today’s office”, KODAK Publication, W-628, 1990.

[12] Watson, I.. “Applying Case-Based Reasoning: Techniques for Enterprise Systems”, Morgan Kaufmann, 1997.

[13] White A., “The Elements of Graphic Design”, Allworth Press, 2002.

[14] Williams R., “The Non-Designer’s Design Book”, Peachpit Press, 2nd Edition, 2004.