DOI: http://dx.doi.org/10.5769/C2011011

Computer Forensic Document Clustering with ART1 Neural Networks*

Georger Rommel Ferreira de Araújo[1], and Célia Ghedini Ralha[2]
(1) Technical-Scientific Unit, Federal Police Department (DPF), Juazeiro do Norte, Brazil,
rommel dot grfa at dpf dot gov dot br
(2) Computer Science Department, University of Brasília (UnB), Brasília, Brazil,
ghedini at cic dot unb dot br

Abstract — **Computer forensic text corpora are usually very heterogeneous. While classification, by file type or other criteria, should be an aid in the exploration of such corpora, it does not help in the task of thematically grouping together documents. Adaptive Resonance Theory (ART) describes a number of self-organizing artificial neural networks that employ an unsupervised learning process and is specially designed to learn new patterns without forgetting what it has already learned, overcoming the important restriction defined by the stability/plasticity dilemma. In this direction, this paper applies the ART1 algorithm (ART with binary input vectors) to thematically cluster documents returned from query tools used with forensic text corpora. Documents that would previously be presented in a disorganized and often long list are thematically clustered, giving the examiner a faster and effective way of obtaining a general picture of document content during forensic examinations. Our experimental results are expressive to validate our approach, achieving high agreement between the clustering solution processed with our software package and the gold standard defined by domain area experts.**

Keywords — ART1; artificial neural networks; computer forensics; document clustering

1. Introduction

Text is very important in forensic examinations. Computer forensic text corpora often consist of a very heterogeneous mix of artifacts, including but not limited to: office suite documents and spreadsheets, PDF files, email, web browser and instant messaging history files, and a huge list of text strings, which are extracted from unallocated and slack space. It is very common to find all of these types of forensic artifacts in computers and discrete media seized by law enforcement. Finding, organizing and analyzing evidence from this diverse mix of artifacts takes more time and effort than the resources available, especially considering the never-ending growth of storage capacity [1].

A. Forensic examinations and forensic text string searches

After evidence is seized or acquired, examination begins. A computer forensic examination is a procedure that is both broad and deep, performed by a specialist that is responsible and legally accountable for handling the evidence, keeping the chain of custody and writing a report with his findings. Depending on the size and complexity of data, an examination may take a long time to complete. Two techniques commonly used to speed up examinations are data reduction [2] and text indexing [3]. Data reduction consists of applying filters during pre-processing to exclude certain portions of data that are known to be safely ignorable, usually by means of querying a hash database, while text indexing consists of building a text index so that files can be searched quickly after the pre-processing phase.

The current process of text string searching using computer forensic tools is basically the same: the forensic examiner inputs search strings and the application returns a number of search results, or hits, for the examiner to peruse. It is not uncommon for text searches to return hundreds to thousands of hits. All hits must be returned, because the application cannot distinguish the relevant ones from the irrelevant. The task of reviewing the hits is subjective in nature and belongs to the domain specialist – the examiner. Thus, it becomes critically important for the application to present hits in such a manner that the examiner may quickly and efficiently review them. Unfortunately, current computer forensic applications at most categorize hits by file type – a process not much more sophisticated than running the tool *grep* on the image file [4]. As storage capacity of seized computers and media begins to surpass the terabyte range, the need for better ways of displaying and navigating search hits becomes even more apparent. A survey distributed to computer forensics experts published in 2010 in an MSc thesis [5] found out that 60.7% of respondents chose "greater efficiency of data searching" as one of the top 3 main changes in forensics tools that would enable the examiner to spend less time on a case. Despite the importance of the research subject, relatively little research has been done to develop efficient searching mechanisms with adequate presentation methods for the exploration of computer forensic text corpora [6, 7, 8, 9,10,11].

B. Document clustering

Text classification or text categorization is the process of assigning text documents to predefined classes or labels. As

* This work was sponsored by the Ministry of Justice of Brazil under the PRONASCI Program.

the classes and the training instances that belong to them are known beforehand, the classification problem is a task of supervised learning [12]. The process of document clustering aims to discover natural groupings (clusters) so that documents within a cluster are similar to one another but are dissimilar to documents in other clusters [13]. Clustering differs from classification in that it assumes no prior knowledge of categories or the existence of a training set, and as such it is a task of unsupervised learning. Clustering may be applied in scenarios where classification is inadequate or impossible. This kind of scenario is commonplace in computer forensic examinations, where the knowledge obtained in a case usually cannot be reused in other unrelated cases.

There exist many clustering algorithms that differ in several aspects, among which one may cite: computational complexity, cluster quality, and sensitivity to the order of presentation of the inputs. While it is desirable to build the best possible clusters, there is no way to define the best clusters, since there are many correct, context-dependent ways to arrange the documents for a given corpus. Clustering quality may be evaluated by means of internal and external criteria. An internal criterion measures intra-cluster similarity (documents belonging to the same cluster should be similar) and inter-cluster similarity (documents belonging to different clusters should be dissimilar). While good scores on an internal criterion are desirable, they do not necessarily translate into good effectiveness in an application [14]. An external criterion, on the other hand, measures how well the clustering solution matches a set of classes produced by human judges, or *gold standard*. Thus, it is desirable for a clustering algorithm to produce clusters that approximate the gold standard classes.

Many clustering algorithms take vectors as input. According to [15], the *vector space model* can be used to represent text documents as vectors of terms. The document collection is represented as a matrix where each document is a row and each term is a column representing a different dimension. In this way, a document collection containing N documents and M terms can be represented by the N x M matrix presented in Table I, where a_{nm} is an attribute that represents the term t_m frequency of term d_n in document . This is also known as the *bag-of-words* model, or simply BOW.

For the purpose of the present work, binary input vectors will be used. The matrix may be modeled as a binary incidence matrix, where a_{nm} is 1 if term t_m occurs in document d_n and is 0 otherwise.

TABLE I. Bag-Of-Words model

	t_1	t_2	⋯	t_M
d_1	a_{11}	a_{12}	⋯	a_{1M}
d_2	a_{21}	a_{22}	⋯	a_{2M}
⋮	⋮	⋮	⋱	⋮
d_N	a_{N1}	a_{N2}	⋯	a_{NM}

Even moderately sized document collections often have several tens of thousands of different terms, with each document containing relatively few terms. The matrix is thus very sparse. This poses a problem for clustering algorithms because considerable resources are spent to represent and perform calculations on a large volume of data that is mostly empty except for a relatively small percentage of terms in each document. This problem is known as the curse of dimensionality, and several reduction techniques have been proposed to deal with it in the domain of document clustering. These techniques aim to reduce the computational complexity of document clustering algorithms, or at least their running time, while keeping the quality and relevance of the output.

C. Search result clustering

A common representation of search results is a simple list the user must scan from top to bottom until she finds the information that addresses her information needs. In the context of computer forensics examinations where each search request may return hundreds to thousands of results, it can be confusing and time-consuming to wade through all of them.

Document clustering can also be applied to a subset of the document collection. *Search result clustering* can be used to cluster the documents that were returned in response to a query. In [16] van Rijsbergen presents the *cluster hypothesis* which states that *"Closely associated documents tend to be relevant to the same requests"*. In other words, if there is a document from a cluster that is relevant to a search request, then it is likely that other documents from the same cluster are also relevant. This is because clustering puts together documents that share many terms [14]. Thus, if the searching results are clustered, similar documents will appear together in coherent groups, which are arguably easier to scan than a disorganized list. If the cluster hypothesis holds, when the examiner finds a cluster where the first few documents are relevant to the query then the whole cluster can be flagged for more detailed analysis; conversely, a cluster where the first few documents are judged irrelevant can be discarded right away.

Search result clustering has been found to offer promising results, both in traditional text-based information retrieval and web-based information retrieval [17,18,19]. This paper focuses on the use of ART1 clustering algorithm to apply to search results returned from queries to computer forensic text corpora. The scenario investigated is that of hard clustering, where each document is a member of exactly one cluster.

The rest of this paper is organized as follows: in Section II we present related research work; in Section III we discuss the Adaptive Resonance Theory with the architecture and the ART1 algorithm; in Section IV we describe our proposed approach to deal with clustering in the computer forensics domain; in Section V we detail the experiments; and finally, in Section VI we present the discussion and conclusions of our work.

2. Related Work

Document clustering of forensic text corpora has been done by researchers using different techniques and models, such as Kohonen's Self-Organizing Maps (SOM) [9] and the k-means algorithm [20]. Beebe and Dietrich in [8] proposed a new process model for text string searches that advocated the use of machine learning techniques, clustering being one of them.

Beebe and Clark in [9] used the Scalable Self-Organizing Map (SSOM) algorithm, which takes advantage of the sparseness of input vectors, to cluster search hits returned from queries to computer forensic text corpora. The results were reported to be promising. Beebe researched the subject further in a PhD thesis [21], where some issues and limitations that warrant further discussion are described.

1. First, the author performed searches at a physical level, completely skipping the file system. This approach is likely to miss documents that are not stored as plain text (Microsoft Office 2007 user files, PDF, among others), or stored in noncontiguous areas of the media being examined.
2. Second, the author also argues against full-text indexing, and states that "*Simply put, the startup cost for index creation is prohibitive – the return on investment is too low to make full text indexing worthwhile*." Both FTK and EnCase, two widely used computer forensic software packages [5], enable full-text indexing by default during pre-processing. This step is known to be resource-intensive, but by no means prohibitive. No evidence of said unfeasibility is presented.
3. Third, the user was required to input the size of the 2-dimensional map that would display the clusters. The work does not mention how to determine or even suggest this value. The author states that "*… the primary purpose of this research was to ascertain the feasibility of clustering digital forensic text string search results and draw general conclusions regarding possible improvement in IR*[1] *effectiveness, studying cluster granularity optimization was out of scope*", and follows stating that "*Future research should experimentally examine optimal map and document vector sizes given various data set and query characteristics*." The present work builds on the ideas discussed on Beebe's research [21].

The approach based on the k-means algorithm required the user to input the number of clusters, k, that the algorithm would create; Decherchi et al. [20] chose $k = 10$, stating that "*… this choice was guided by the practical demand of obtaining a limited number of informative groups*," and clustered the whole forensic text corpus they chose for their experiment. After the clustering was done, the words considered to be the most descriptive among the twenty most frequent words present in the documents belonging to each cluster were assessed and the results were reported to range from "interesting" to "extremely interesting". The authors did not discuss the technical details of the implementation.

As far as we are concerned, no other work related to document clustering of forensic text corpora was found; although there is a very large number of published works about document clustering. L. Massey published several research papers [22,23,24] and a PhD thesis [25] on the subject of document clustering using the ART1 algorithm.

3. Adaptive Resonance Theory

Adaptive Resonance Theory (ART), inspired by human cognitive information processing, was introduced by Grossberg in 1976 [26]. It describes a number of self-organizing artificial neural networks that employ an unsupervised learning process, and one of its main goals is to overcome the stability/plasticity dilemma. The stability/plasticity dilemma lies in that a system is desired which is able to learn new patterns without forgetting what it has already learned. A number of other previous neural network models are not plastic because they cannot learn from new patterns after the network is first trained. They are also not stable because even though they can be retrained from scratch to process new patterns, they do so at the cost of quickly forgetting old knowledge.

ART neural networks, on the other hand, are plastic because they can dynamically learn new patterns even after the network has stabilized, and are also stable since they preserve knowledge about past input patterns as new ones are presented. Each cluster corresponds to an output neuron, and the algorithm creates and updates them on-the-fly as required by the input data, meaning the network is self-organizing. These properties make ART neural networks suitable for incremental clustering of data. The present work focuses on document clustering with ART1, an ART neural network designed to work with binary input vectors.

The foundations of ART are laid out in detail in a large number of publications by Grossberg, Carpenter and other researchers [27,28]. A clear and concise introduction that skips the mathematical underpinnings of ART and concentrates on the architecture of the network and the operation of the algorithm can be found in [29].

A. ART1 architecture

Fig. 1 presents a typical ART architecture using a block diagram. There are two main components, the *attentional* and *orienting* subsystems. The attentional subsystem contains, among other components, two layers of neurons, F_1 and F_2. The *comparison layer* F_1 has N input neurons, and the *recognition layer* F_2 has M output neurons. N is the input size, i.e. the number of input patterns (in this case, documents). M is the number of clusters produced by the algorithm, and

[1] Information Retrieval

is calculated dynamically. Neurons are fully connected with both feed-forward and feedback weighted links. The orienting subsystem contains the reset layer for controlling the overall dynamics of the attentional subsystem.

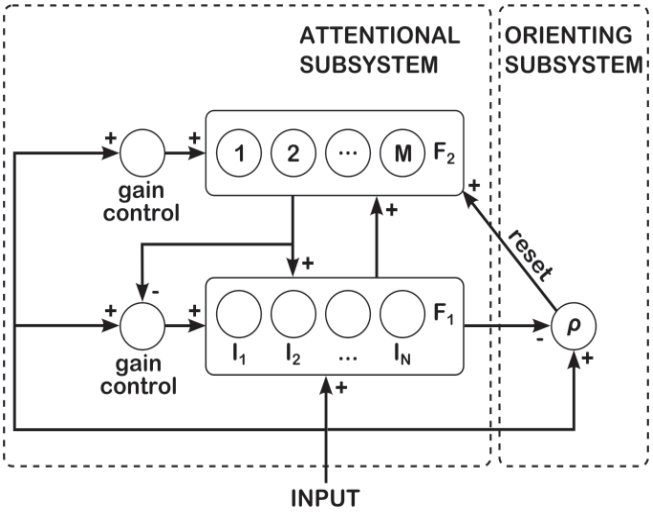

Figure 1. ART1 architecture [28]

Binary input patterns are presented to the F_1 layer, and the feed-forward links present the inputs to the F_2 layer. The feedback links from F_2 to F_1 store the prototype vectors, i.e. the vectors that represent the clusters embodied by the output neurons. The output neurons (clusters) in the F_2 layer compete for activation. The neuron with maximum activation takes the value 1 and inhibits the others, which take the value 0. If the input presented in F_1 and the winning neuron in F_2 "match", i.e. share enough dimensions in common, *resonance* occurs; *reset* occurs otherwise and the search is repeated. If no matching prototype vector is found, a new cluster is created based on the input vector. An input vector and a prototype vector are said to match if the *vigilance test* is passed. Vigilance (ρ) is a dimensionless parameter that dictates how close an input must be to a prototype vector for resonance to occur.

B. ART1 algorithm

The original work about ART1 describes its architecture and operation by means of differential equations, and does not supply a reference implementation [28]. Fig. 2 presents the *Cluster-ART-I* algorithm described in [30] and cited in [27,31], with additional remarks for clarification. This algorithm is used in this paper as the basis of the software package developed and validated in the experiments. The Cluster-ART-I algorithm will be referred to simply as ART1 algorithm from now on.

The ART1 algorithm groups binary input vectors into categories or clusters. Each cluster has a prototype vector, and each input vector is classified into the cluster that has the nearest prototype vector. The algorithm will create as many clusters as required by data. The number of clusters and their size depend on two parameters, β and ρ.

β is the *choice parameter*, a small positive number that is added to the denominator in step 2 of the algorithm in order to avoid division by zero should it happen that $||T_i||_1 = 0$. The limit $\beta \to 0$ is called the *conservative limit* because small values of β tend to minimize recoding, i.e. updating of the prototype vectors, during the learning process [27]. β is used in the *category choice function* described in step 2 and also in the distance test in step 3 of the algorithm.

0. Start with zero cluster prototype vectors: the set P of prototype vectors is {}.
1. Let I = next input vector. Let P', the set of candidate prototype vectors, be equal to P.
2. Find the closest prototype vector (if any). Call this prototype vector T. To find T is to find $i \in P$ to maximize
$$\frac{T_i \cdot I}{\beta + \|T_i\|_1}$$
for some $\beta \ll 1$.
$\|T_i\|_1$ counts the number of 1s in T_i and acts as a tie-breaker, favoring smaller magnitude prototype vectors over larger prototype vectors which are subsets of them (have 1s in the same places) when input vector I matches them equally well ($T_i \cdot I = T_{i'} \cdot I$).
3. If $P' = \{\}$, or if T_i is too far from I:
$$\frac{T_i \cdot I}{\beta + \|T_i\|_1} < \frac{I \cdot I}{\beta + n}$$
, then create a new cluster, j, and set $T_j = I$.
Set $P = P \cup \{j\}$. Output j. Then go to step 1.
n is the number of dimensions (dimensionality) of I.
3′. If T_i does not match I, in other words,
$$\frac{T_i \cdot I}{I \cdot I} < \rho$$
, $0 < \rho \leq 1$, then set $P' = P' - \{i\}$ and go to step 2.
4. Otherwise (T_i passes the tests in steps 3 and 3′ and thus is close enough to I in both senses), then update T_i: $T_i \leftarrow T_i \cap I$. Output i. Go to step 1.

Figure 2. ART1 algorithm [30]

ρ is the *vigilance* parameter, $0 < \rho \leq 1$, that tests the similarity between the input and prototype vectors in step 3' of the algorithm and directly influences the number of clusters the algorithm creates, as well as their size. Low vigilance leads to coarse-grained generalization and abstract prototype vectors that represent many input vectors, the result being fewer clusters containing each a larger number of input vectors. High vigilance leads to fine-grained generalization and prototypes that represent fewer input vectors, the result being more clusters containing each a smaller number of input

vectors. Though ART1 is unsupervised, the user is allowed a limited amount of supervision by means of tuning the vigilance parameter. Such tuning is useful when the number and size of the clusters is considered too small or too large.

It is stated in [27] that both Fuzzy ART (another member of the ART family) and ART1 take three parameters: a choice parameter α, a learning rate β, $0 < \beta \leq 1$, and vigilance ρ; although [30] mentions only two, choice parameter β and vigilance ρ. Even though each work describes a different set of parameters for the ART1 algorithm, it is the same algorithm. This is because the ART1 algorithm described in [27] implements the *fast learning mode* described in [28], where the learning rate β is set to 1, and uses the symbol β for the choice parameter instead of α.

4. Proposed Approach

The proposed approach consists of the development of a software package and a validation method used during the experiments with a real world case.

A. Software package

Two command line programs were produced. The first one was the *indexer*, whose task was to traverse a file system exposed from a mounted image file looking for: (i) all Microsoft Office user files (.doc, .docx, .xls, .xlsx, .ppt, .pptx, .pps, .ppsx); (ii) text files (.txt and .rtf) files; (iii) PDF files (.pdf); (iv) HTML and XML files (.htm, .html and .xml). The program also runs the following pre-processing steps:

- Extract text from structured documents.
- Look for text in Portuguese; documents where no text in Portuguese was found were discarded.
- Remove stopwords from the extracted text.
- Tokenize and stem the extracted text.
- Build an index with the metadata from each file and its respective extracted text for later query and retrieval.

The second program was the *searcher*, whose task was to query the index and present the results i.e. the documents that matched the query. Upon execution, the program would return up to 1,000 documents in the index that matched the query expression, cluster them together with the ART1 algorithm, and then write a number of HTML files presenting all clusters and their respective documents along with the most frequent 20 terms (based in [20]) present in the documents belonging to each cluster, which allow to examiner to obtain a general picture of the documents in each cluster. Clusters are numbered sequentially and have no labels other than their numbers. The searcher program was written in Java and was based on the source code available in [31]. The program took three parameters: the first and the second were mandatory – case number and query expression, while the third was an optional vigilance value.

After the searcher program is executed and retrieves the documents that match the query expression and clusters them, the clustering solution must be compared to the gold standard classes defined by the domain experts (as presented in Section D of IV).

B. Comparison of clusterings

This description in based on [32]. Let $D=\{d_1,d_2,\ldots,d_n\}$ be the set of documents N matching the query expression, and $U=\{u_1,u_2,\ldots,u_R\}$ and $V=\{v_1,v_2,\ldots,v_C\}$ two partitions of the documents in D such that $\bigcup_{i=1}^{R} u_i = D = \bigcup_{j=1}^{C} v_j$ and $u_i \cap u_{i'} = \emptyset = v_j \cap v_{j'}$ for $1 \leq i \neq i' \leq R$ and $1 \leq j \neq j' \leq C$. Partition U has R subsets (clusters) and represents a clustering solution, and partition V has C subsets (classes) and represents the gold standard. Let d_1 and d_2 be a pair of documents chosen from D. The total number of possible combinations of pairs in D is $\binom{n}{2}$, and the pairs can be of four different types:

- a – objects in a pair are placed in the same cluster in U and in the same class in V
- b – objects in a pair are placed in the same cluster in U and in different classes in V
- c – objects in a pair are placed in different clusters in U and in the same class in V
- d – objects in a pair are placed in different clusters in U and in different classes in V

A contingency table can be computed to indicate overlap between U and V.

TABLE II. Contingency Table for Comparing Partitions U and V

$U \backslash V$	V_1	V_2	$\cdots$	V_C	Sums
U_1	n_{11}	n_{12}	$\cdots$	n_{1C}	a_1
U_2	n_{21}	n_{22}	$\cdots$	n_{2C}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
U_R	n_{R1}	n_{R2}	$\cdots$	n_{RC}	a_R
Sums	b_1	b_2	$\cdots$	b_C	N

In Table II n_{ij} represents the number of documents that were clustered in the ith subset (cluster) of partition U and in the jth subset (class) of partition V. The values of a, b, c and d can be computed from the n_{ij} values present in Table II.

Intuitively, a and d can be interpreted as indicators of agreement between U and V, while b and c can be interpreted as indicators of disagreement. Several indices based on these four types of pairs have been proposed in the literature. This study uses an information theoretic based measure, the Normalized Mutual Information (NMI).

C. Normalized Mutual Information

The Normalized Mutual Information (NMI) is a measure with a strong background rooted in Information Theory that gives a sound indication of the information shared between a pair of clusterings [33]. Let $I(U,V)$ be the Mutual Information (MI) between the random variables described by clusterings

U and V, and $H(U)$ be the entropy[2] of the random variable described by the cluster labeling U. The Mutual Information, or MI, defined by $I(U,V)$ is a metric or distance. As it has no upper bound, a normalized version that ranges between 0 and 1 is desirable [33]. The NMI is defined as presented in Equation 1, which needs to be estimated from the sampled quantities provided by the clusterings. To meet this need, the normalized mutual information estimate $\phi^{(NMI)}$ presented in Equation 2 is used [33].

$$NMI(U,V) = \frac{I(U,V)}{\sqrt{H(U)\times H(V)}} \quad (1)$$

$$\phi^{(NMI)}(U,V) = \frac{\sum_{i=1}^{R}\sum_{j=1}^{C} n_{ij}\times\log\left(\frac{n_{ij}\times N}{a_i\times b_j}\right)}{\sqrt{\left(\sum_{i=1}^{R} a_i\times\log\left(\frac{a_i}{N}\right)\right)\times\left(\sum_{j=1}^{C} b_j\times\log\left(\frac{b_j}{N}\right)\right)}} \quad (2)$$

The NMI has a fixed lower bound of 0, and a fixed upper bound of 1. It takes the value of 1 when two clusterings are identical, and takes the value of 0 when they are independent, i.e. they do not share any information about each other.

D. Validation method

The search results for each query expression are classified by a human expert, i.e. a domain specialist, to form the classes that represent the gold standard. Then the clusters produced by the software package are compared to the classes, and $\phi^{(NMI)}$ is computed. Since $\phi^{(NMI)}$ is a quantitative measure, a subjective evaluation will also be performed by another domain specialist, who will assess the produced clusters to define whether they are "good" and "useful"; in other words, judge whether the algorithm managed to group similar documents and give a good general picture of their contents.

5. Experiments

An image file acquired from the hard drive of a computer seized by a law enforcement agency during the course of an actual investigation was processed. The computer contained a single hard drive and was seized while executing a search warrant for collecting evidence of fraud in public purchasing contracts. The hard drive contained a single NTFS file system.

A. Document count

Table III presents the number of documents which were used in our experiments, categorized by document type.

TABLE III. Total Number of Documents

File type	*# of documents*
Microsoft Office user files (.doc, .docx, .xls, .xlsx, .ppt, .pptx, .pps, .ppsx)	701
Text files (.txt,.rtf)	79
PDF files (.pdf)	400
HTML and XML (.htm, .html,.xml)	2,818

B. ART1 parameters

As presented in Section B of III, the choice parameter β was set to the small value of 0.01 in step 2 of the algorithm as described in Fig. 2.

The default vigilance parameter was set to $\rho = 500/M$. This value was set as a starting point as discussed in [25], where it is suggested that the value for the *minimal useful vigilance* (ρ_{min}) as $\rho_{min}=1/M$ for a document collection with a maximum dimensionality of 2,600, and in practice much less (500 to 800). Such a value is not adequate for the purpose of this study because no dimensionality reduction techniques other than stopword removal and stemming are performed, which causes search results returned by queries to possess high dimensionality (>3,000). Although the default vigilance was calculated at runtime, it was only used as a suggestion and the user could quickly review the produced clusters and run the software again as many times as desired, supplying a value for vigilance, in order to find clusters that the user considers more adequate.

C. Results

Table IV lists the results. Company and individual names were masked to protect their rights. Tables V and VI are the contingency tables for each query expression. Cells whose value is 0 were left blank to improve legibility.

TABLE IV. Results

Query expression	*# of results*	*# of classes*	*# of clusters*	$\phi^{(NMI)}$	ρ
Person#1	90	15	15	0.831	0.05
Company#1	325	23	26	0.7473	0.08

[2] A clear and concise review of the fundamental concepts involved can be found in [34].

TABLE V. Contingency Table for Query Expression "Person#1"

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Sums
1							23									23
2				1	1									3		5

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Sums
3						1									1	2
4	1															1
5										2						2
6						4							1			5
7				7									1			8
8										15						15
9				2								3				5
10													10			10
11													5			5
12			1							1						2
13		1							2	1						4
14											1		1			2
15								1								1
Sums	1	1	1	10	1	5	23	1	2	19	1	3	18	3	1	90

TABLE VI. Contingency Table for Query Expression "Company#1"

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	Sums
1													86									4		90
2								4	1															5
3														2										2
4								3													1			4
5								2									1							3
6						1														1				2
7	1							2																3
8		1						1															1	3
9								21							1									22
10								1											6					7
11											1	2												3
12				1				2																3
13			80		1																			81
14								6												2			1	9
15								1											2	1				4
16																				10				10
17							1	16												6				23
18						2												3						5
19										2										1				3
20								11																11
21														4										4
22																		1		1				2

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	Sums
23								7								1		1		2				11
24						1		5										1	2					9
25																				2				2
26																		4						4
Sums	1	1	80	1	1	4	1	82	1	2	1	2	86	6	1	1	1	10	10	26	1	4	2	325

6. Discussion and Conclusions

The results of the experiments raise a number of topics for discussion and present opportunities for future work. Some conclusions are also drawn.

A. Discussion

At first, the searcher program generated 20 clusters for query expression "Person#1" with automatically calculated vigilance ρ=0.107089. In order to find out how close the ART1 algorithm could get to the gold standard of 15 classes for this particular case and query expression, vigilance was decreased until the searcher program output 15 clusters. Table IV presents the final vigilance of 0.05 which resulted in $\phi^{(NMI)}$=0.831. Considering that the upper bound for NMI is 1, this can be considered a good result – there is high agreement between the clustering solution and the gold standard. This agreement is portrayed in Table V, where the dispersion of documents between clusters and classes is arguably small.

In the case of query expression "Company#1", tuning of the vigilance was based on the subjective evaluation by the domain specialist of document labels (file names) in each cluster. $\phi^{(NMI)}$ is lower than with query expression "Person#1", but still high at 0.7473. As for the subjective evaluation, the domain specialist considered the generated clusters "good" and "useful".

Even in the cases where the documents belonging to a particular class were dispersed among several clusters, they were not randomly fragmented between clusters, but rather were mostly concentrated on a small number of clusters. One possible explanation for this is that the documents, although belonging to the same class, were not considered similar enough to be included in the same cluster by the clustering algorithm. This was verified to be true when two documents that were classified in class "CONTRACTS" had few words in common other than the word "CONTRACT" in their first page and file names; the specialist would put them in the same class, even though their content was very different.

The domain specialist complained that the searcher program did not give descriptive labels to the clusters, but rather just displayed a number and the labels (file names) of the documents. Cluster labeling is a research topic by itself and was not addressed in this work.

Many of the returned documents were structured (.doc, .docx) rather than plain text (.txt, .html). These would possibly not be found if the approach described in Section II, item 1 was followed.

B. Conclusions

Unfortunately, we found little previous research in document clustering of forensic text corpora to compare results, but the work of [21] compared to our results shows that:

- It is possible to generate good clusters without the need to specify runtime parameters (vigilance was calculated at runtime and could be tuned if so desired).
- It is not enough to cluster only results obtained at the physical level.

Nevertheless, there is much more investigation to be done in this area. The use of the ART1 algorithm was found to be feasible and useful for clustering of computer forensic documents, but the algorithm should still be improved considering the processing time efficiency since the document-term matrix is usually very sparse.

This work presented research that is justified in the basis that it can be confusing and time-consuming to wade through hundreds to thousands of search results returned from queries to computer forensic text corpora. Document clustering in computer forensic examinations using ART1 neural networks was proved to allow forensic examiners to quickly and effectively obtain a general picture of document contents, leveraging van Rijsbergen's cluster hypothesis.

Although, apart from the use of different techniques, nothing can compare to thorough and careful review of each search result by a human examiner, document clustering can help provide quick insights into the contents of seized media. Such a capability can be useful, for example, when the goal is to gather intelligence quickly prior to performing a complete examination.

As future work we may cite the use of soft and hierarchical clustering, the generation of high quality cluster labels, and the comparison with other clustering algorithms such as SOM, k-means and Expectation-Maximization (EM).

Acknowledgments

The authors are very grateful to Prof. Maria das Graças Volpe Nunes, PhD, for kindly sharing the stopword list used

in several research projects at the Núcleo Interinstitucional de Linguística Computacional (NILC) at the Univerity of São Paulo. Thanks are also due to Luís Filipe da Cruz Nassif for suggesting the use of the NMI.

References

[1] N. Beebe and J. Clark, "Dealing with terabyte data sets in digital investigations," Advances in Digital Forensics 194, pp. 3-16 (2005).

[2] N. Beebe and J. Clark, "A hierarchical, objectives-based framework for the digital investigations process," Digital Investigation, Volume 2, Issue 2, June 2005, pp. 147-167 (2005).

[3] C. Johansson, "Computer Forensic Text Analysis with Open Source Software," MSc thesis, Dept. of Software Engineering and Computer Science, Blekinge Tekniska Högskola (2003).

[4] D. Forte, "The importance of text searches in digital forensics," Network Security, 2004, pp. 13 – 15 (2004).

[5] N. Kuncik, "Introducing Data Mining to Digital Forensic Investigation Process," MSc thesis, UCD School of Computer Science and Informatics, College of Engineering Mathematical and Physical Sciences, University College Dublin, Ireland (2010).

[6] H. Jee, J. Lee, and D. Hong, "High Speed Bitwise Search for Digital Forensic System," World Academy of Science, Engineering and Technology, 32, pp. 104-107 (2007).

[7] J. Lee, "Proposal for Efficient Searching and Presentation in Digital Forensics," Third International Conference on Availability, Reliability and Security, IEEE Computer Society, 0, pp. 1377-1381 (2008).

[8] N. Beebe and G. Dietrich, "A new process model for text string searching," in Research advances in digital forensics III, Shenoi S, Craiger P, Eds.. Norwell: Springer; 2007. pp. 73–85.

[9] N. Beebe and J. Clark, "Digital forensic text string searching: Improving information retrieval effectiveness by thematically clustering search results," in Digital Investigation, September 2007, vol. 4 (suppl. 1) (2007).

[10] M. Schwartz and L. M. Liebrock, "A Term Distribution Visualization Approach to Digital Forensic String Search," in Proceedings of the 5th international workshop on Visualization for Computer Security (VizSec '08), John R. Goodall, Gregory Conti, and Kwan-Liu Ma (Eds.). Springer-Verlag, Berlin, Heidelberg, pp. 36-43. 2008.

[11] M. Schwartz, C. Hash, and L. M. Liebrock, "Term distribution visualizations with Focus+Context," in Proceedings of the 2009 ACM symposium on Applied Computing (SAC '09). ACM, New York, NY, USA, pp. 1792-1799. 2009.

[12] F. Sebastiani, "Machine learning in automated text categorization," ACM Computing Surveys, 34(1), pp. 1–47 (2002).

[13] N. O. Andrews and E. A. Fox, "Recent Developments in Document Clustering," unpublished, http://eprints.cs.vt.edu/archive/00001000/ (2007).

[14] C. D. Manning, P. Raghavan, and H Schütze, Introduction to Information Retrieval. Cambridge University Press, 2008.

[15] G. Salton. Automatic Text Processing. Addison-Wesley, 1989.

[16] C. J. van Rijsbergen, Information Retrieval. Butterworths, London, 2nd edition. 1979.

[17] A. Leuski, "Evaluating document clustering for interactive information retrieval," in Tenth international conference on information and knowledge management. Atlanta, Georgia: ACM Press; 2001.

[18] A. Leuski and J. Allan, "Improving interactive retrieval by combining ranked lists and clustering in RIAO," College de France; 2000.

[19] H. Zeng, Q. He, Z. Chen, W. Ma, and J. Ma, "Learning to cluster web search results," in Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '04). 2004. ACM, New York, NY, USA, pp. 210-217.

[20] S. Decherchi, S. Tacconi, J. Redi, A. Leoncini, F. Sangiacomo, and R. Zunino, "Text Clustering for Digital Forensics Analysis," in Journal of Information Assurance and Security, No. 5, pp. 384-391, 2010.

[21] N. L. Beebe, "Improving information retrieval effectiveness in digital forensic text string searches: clustering search results using self-organizing neural networks," PhD thesis, Department of Information Systems and Technology Management, College of Business, The University of Texas at San Antonio, 2007.

[22] L. Massey, "Determination of Clustering Tendency With ART Neural Networks," in Proceedings of 4th Intl. Conf. on Recent Advances in Soft Computing, Nottingham, U.K., 12 & 13 December 2002.

[23] L. Massey, "On the quality of ART1 text clustering," Neural Networks(16)5-6, pp.771-778, 2003.

[24] L. Massey, "Real-World Text Clustering with Adaptive Resonance Theory Neural Networks," in Proceedings of 2005 International Joint Conference on Neural Networks, Montreal, Canada, July 31- August 4, 2005.

[25] L. Massey, "Le groupage de texte avec les réseaux de neurones ART1," PhD thesis, Faculté d'ingénierie du Collège militaire royal du Canada, 2005.

[26] S. Grossberg, "Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors," Biological Cybernetics, 23, pp. 121–134, 1976.

[27] G. A. Carpenter, S. Grossberg, and D. B. Rosen, "Fuzzy ART: Fast Stable Learning and Categorization of Analog Patterns by an Adaptive Resonance System," Neural Networks, 1991. Elsevier Science Ltd. Oxford, UK, UK, Volume 4 Issue 6, pp.759-771.

[28] G. A. Carpenter and S. Grossberg, "A massively parallel architecture for a selforganization neural pattern recognition machine," Computer Vision, Graphics, and Image Processing, vol 37, pp. 54-115, 1987.

[29] L. Heins and D. Tauritz, "Adaptive resonance theory (ART): An introduction," unpublished, http://web.mst.edu/~tauritzd/art/art-intro.html (1995).

[30] B. Moore, "ART 1 and Pattern Clustering," in Proceedings of the 1988 Connectionist Models Summer School, pp. 174-183, 1988.

[31] T. Hudík. Tool: ART software package. http://users.visualserver.org/xhudik/art/, accessed 07/23/2011.

[32] J. Santos and S. Ramos, "Using a clustering similarity measure for feature selection in high dimensional data sets," Intelligent Systems Design and Applications (ISDA), 2010 10th International Conference on, 2010, pp. 900-905

[33] A. Strehl, J. Ghosh, and C. Cardie (2002), "Cluster ensembles - a knowledge reuse framework for combining multiple partitions," Journal of Machine Learning Research, 3: pp. 583–617.

[34] N. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: is a correction for chance necessary?," in Proceedings of the 26th Annual International Conference on Machine Learning (ICML '09)