A Case-based Approach to Content Analysis in Cross-Domain Information Sharing Environments

Thomas Reichherzer, Badri Lokanathan, Ashwin Ram
Enkia Corp.[1]
Atlanta, GA, USA

***Abstract*—Information sharing plays a critical role in government and industry alike. While information sharing is desirable, it must be controlled to prevent leakage of sensitive content to unauthorized individuals and organizations. Reliable human review (RHR) is the preferred method to apply security policies to decide upon the release of information. Unfortunately, time-critical situations and the ever increasing volume of electronic documents make RHR no longer feasible. This paper describes methods that combine case-based reasoning (CBR) with natural language processing (NLP) techniques to learn security policies from marked up data in unstructured text documents. The learned policies can then be applied to unmarked text documents for classifying content according to the policies. The methods have been evaluated using data selected from the IMdb database for classifying text documents.**

I. INTRODUCTION

The ability to identify and share information in a "just enough" manner is critical in defense, intelligence, and industry settings. In the military and intelligence communities, analysts and decision-makers are responsible for reading, digesting, and sharing critical information with counterparts in their own agencies, in other US government agencies, and even in other countries. A key aspect of Cross-Domain Information Sharing (CDIS) is reliable human review (RHR): the review of documents, emails, and other information being shared to ensure adherence to non-disclosure policies across multiple security domains.

Information review is performed by document authors as well as review officers (e.g., foreign disclosure officers (FDOs)) – to provide the necessary checks and balances. It is a time-intensive process requiring significant human effort and expertise. With information overload reaching record proportions, human analysts are overwhelmed and unable to keep up with the sheer volume of data. This problem is magnified by the increasing complexity of information sharing policies, be they national disclosure policies for classified information in the government or human resources policies in a large company. New tools are therefore needed to assist analysts with the review process.

Primary tools available today to support information sharing include markup tools such as "dirty word checkers" or Bayesian text classifiers [7], to determine if content is sensitive. Markup tools enable humans to highlight certain information (e.g., top secret information embedded in a secret document), but human effort is necessary to read and identify this information. Dirty word checkers use keyword lists to flag sentences or paragraphs for human review; Bayesian classifiers label documents or text excerpts with categories using computed probabilities of word occurrences across document libraries. Both approaches are very limited in their ability to identify relevant information in accordance with security policies. Significant effort is required to develop and maintain dirty word lists, yet the lists are never complete due to inherent ambiguity in word meanings. Furthermore, such lists can never capture all the nuances of non-disclosure policies, which are often broad and high-level. For example, it would be impossible to develop a complete and consistent list of all the words that might constitute "order of battle information," required to capture just one of the categories of our national disclosure policy (NDP).

This paper discusses automated methods for content review and classification and their implementation and evaluation in a prototype system of a text classifier. The methods combine case-based reasoning (CBR), a machine learning technique, with natural language processing (NLP) for decision making. The key insight behind our research is to leverage the same flood of data that is overwhelming human reviewers and to use it to feed a data-hungry Artificial Intelligence (AI) engine for training purposes. The engine parses existing marked up information in the text documents to build libraries of marked up sample cases that captures the implicit judgment of reviewers in classifying information in text documents as sensitive. The sample cases can then be applied to markup new text documents in accordance with the original reviewer's judgment on the classification of content. The construction of a case base can be easily bootstrapped using existing classified and marked up data. Furthermore, feedback from reviewers can be used to continuously refine and revise the engine's classification knowledge allowing it to self-learn and improve with time.

II. THE ARCHITECTURE FOR ASSISTED RELIABLE REVIEW

From the standpoint of solving the information-sharing problem, the general idea is to train a text classifier using marked up text documents from a document library. The markup describes the sensitivity content in accordance with the security policies that govern the release of information across different security domains. From the marked up data, the classifier builds a knowledge base by breaking text into sentences and storing each sentence as a unique case along with policy information for reuse in subsequent classification tasks. When sufficiently trained, the knowledge base con-

[1] The author's work described in this paper was completed at Enkia Corp. The authors have since moved on to positions elsewhere.

tains sample data of content classifications that are considered sensitive as judged by RHR. Since policies apply differently across different release domains, each knowledge base essentially captures classification knowledge with respect to a particular release domain. The knowledge base can be applied subsequently to (1) decide upon classification of information in new text documents and (2) filter or redact the information in preparation for their release.

Figure 1 illustrates the complete architecture of a prototype system that implements the text classifier. The architecture is broken into a text analytical and case construction component as well as a case insertion and retrieval component that are traditionally associated with case-based reasoning engines. The figure also illustrates the data flow for training the classifier as well as for applying it to new and unmarked text documents. In the training phase, the classifier uses marked up text documents as input and generates a case library as output to capture a set of security policies that classify information. In the application phase, the text classifier uses any unmarked text document and cases from the case library as input to generate mark-up suggestions in the document.

The text analytical component first segments the marked up text into a list of sentences before it performs various natural-language processing operations to generate new sentences. Details of the processing steps are described below. The new sentences are then converted into feature vectors [8]. The case-building component uses the label of the markup that identifies the sentences as sensitive and combines it with the feature vector to create a case for storage in the case base. Subsequently, the text classifier can retrieve cases using feature vectors compiled from unmarked sentences to decide upon their mark-up. This task is performed by the CBR retrieval and marker/classifier component in the architecture.

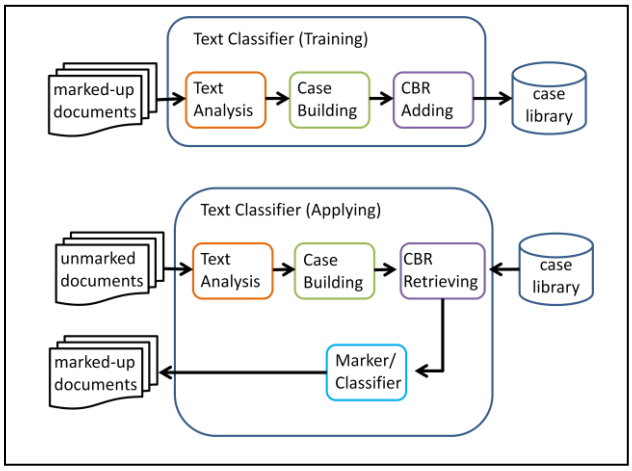

Figure 1. An Architecture for Assisted Reliable Review

III. Case-based Methods for Text Classification

The conversion process of the text analytical component applies natural language processing techniques such as pronoun reference resolutions and shallow-semantic parsing ([4][6]) to build machine representations from marked up sentences. The pronoun resolution step replaces pronouns by their corresponding noun phrases before the shallow-semantic parser breaks the resulting sentences into constituents to build a case representation. The constituents describe the different semantic roles of words that occur within a sentence. For example, the semantic constituents of a sports event may describe activities and actors and the location where the activities take place. In essence, semantic role parsing attempts to label a given sentence such that WHO did WHAT to WHOM, WHEN, WHERE, etc. have their respective answers. After using a semantic-role parser, features are extracted from the parse result to build the feature vector with each constituent representing a separate component in the vector. In essence, the vector captures an example of sensitive information at the sentence-level in the case base and can be used to label semantically similar sentences.

Case-based reasoning is a problem solving methodology that retrieves and adapts previously stored cases to solve similar problems ([1],[2]). Our prototype text classifier utilizes a feature vector, compiled from an unmarked sentence as a problem description that may then be used for choosing similar cases from a case library. From the chosen cases, the classifier derives a label to classify the sentence. If no cases are similar, no label is applied and the sentence is considered non-sensitive. A similarity threshold is used to decide upon similarity between problem description and cases in the case base.

In a bottom-up approach, the classifier examines individual sentences within paragraphs for their sensitivity to assess the sensitivity level of a paragraph. Highly sensitive content supersedes less sensitive content in the classification of paragraphs. Finally, the classifier assigns the entire document a classification label based on the policy and classification of paragraphs. For example, if a paragraph contains a sentence labeled according to policy A and a few sentences labeled according to policy B and if furthermore policy A characterizes top secret and policy B only secret material, then the entire paragraph would be classified as top secret because of the occurrence of the more sensitive top secret sentence contained therein.

For selecting cases from the case base, the CBR component uses a k-nearest neighbor algorithm that measures similarity between the problem description and each case in the case library to select the k "closest" cases as candidates for a solution based on a similarity metric. Finally, from the selected cases, the decision on what label to assign to a given sentence is determined via majority voting. The label that has the largest frequency among the selected cases is chosen to mark the sentence.

The k-nearest neighbor selection process relies on a similarity metric that measures the distance between a problem description and cases in the case library to select closely matching cases. As cases are derived from sentences, intuitively, the distance between any two cases must consistently reflect the degree of semantic similarity between the two corresponding sentences. For example, consider the following three sentences:

- S_1: John bought a house.
- S_2: John rented an apartment.
- S_3: A truck hit Mary's car.

S_1 and S_2 have to do with living accommodations in which John is the actor. S_3 has nothing to do with living accommodations and involves an object as an actor. Intuitively, the semantic distance between S_1 and S_2 is shorter than between either S_1 and S_3 or S_2 and S_3.

For measuring the similarity between the sentences, each sentence must be broken into constituents such as actors, actions, and objects so that the individual constituents can be compared. The assumption is that sentences are semantically more similar if their constituents match. Formally, consider two sentences S_i and S_j. We can express S_i and S_j in terms of feature vectors using the following notation:

$S_i = <f_{i1}, f_{i2}, f_{i3}, f_{i4}, \ldots, f_{im}>$
$S_j = <f_{j1}, f_{j2}, f_{j3}, f_{j4}, \ldots, f_{jm}>$

Here, f_{qk} are features from a sentence S_q, for any q = i, j and for any k between 1 and m. The value m corresponds to the numbers of features in a sentence that are generated by a shallow-semantic parser. Also, since a feature is just a collection of words, any given pair of features f_{ik} and f_{jk} may be expressed as follows:

$f_{ik} = [w_{ik1}, w_{ik2}, \ldots, w_{ikr}]$
$f_{jk} = [w_{jk1}, w_{jk2}, \ldots, w_{jkq}]$

In this expression, feature f_{ik} has *r* feature words, while f_{jk} a total of *q* feature words. Let $D(S_i,S_j)$ denote the similarity between any given pair of sentences and $\sigma(f_{ik},f_{jk})$ the similarity between two corresponding features in S_i and S_j. Then the similarity between two sentences can be measured as:

$$D(S_i,S_j) = k_1\sigma(f_{i1},f_{j1}) + k_2\sigma(f_{i2},f_{j2}) + \ldots + k_m\sigma(f_{im},f_{jm}),$$

$$\sigma(f_{ik},f_{jk}) = \lambda(w_{ik1},w_{jk1}) + \lambda(w_{ik1},w_{jk2}) + \ldots + \lambda(w_{ik1},w_{jkq}) + \lambda(w_{ik2},w_{jk1}) + \ldots + \lambda(w_{ik2},w_{jkq}) + \ldots + \lambda(w_{ikr},w_{jkq}).$$

The λ function is a word similarity metric and the σ function is in turn a linear function of the similarity between words in two features. For the implementation of the similarity metric in the prototype system, the λ function is the word pair similarity given by Dekang Lin's thesaurus [5]. Lin's similarity measure between any two given words is based on the probability distribution of words within a large text corpus. The corpus from which the thesaurus was derived included approximately 22 million words that occurred in articles selected from the Wall Street Journal and the San Jose Mercury. Lin's measure takes into consideration how often a given pair of words are used in a similar way in the corpus, e.g., as the subject of the same verb or as the direct object of the same verb. The more these relationships are observed between any given pair of words, the higher their similarity value.

IV. Experiments and Results

The performance of the classifier was evaluated in several experiments using data collected from the Internet Movie database (IMdb) involving movies rated PG, PG-13, and R. The collected data consisted of movie descriptions that were manually marked up by labels from 6 different categories including general violence, graphic violence, nudity, drug use, dark topics, and sexual content. The mark-up information was compiled into a list of sentences along with category labels. The individual sentences and category labels were then used to train the text classifier resulting in a case base with a total of 1946 cases from the different categories. Table 1 depicts the distribution of categories of cases.

Category	**Number of Cases**
General violence	65
Graphic violence	730
Nudity	498
Drug use	349
Dark topics	298
Sexual content	6

Table 1: Distribution of cases covering categories of mark-up data.

Following the training phrase, tests were performed to measure accuracy of text classification. The tests involved classifying sentences from the training set under two different conditions by varying the size of the case base generated from the training set. Under Condition A, all cases from the case base were used to test classification of sentences, under Condition B only 90% of the cases were available for the classification task, with 10% of the cases randomly selected and removed prior to testing. For Condition B, a total of 10 tests were performed to compute an average result of classification. Table 2 shows the results under both conditions.

Test	**Condition A**	**Condition B**
Number of active cases in case base.	1946	1752
Number of documents processed	28	28
Total number of text markups in training set	1017	1017
Total number of correct classifications.	707	680
Total number of missed classifications.	16	16

Table 2: Results from reclassifying marked up text after the classifier was fully trained.

A total of 16 out of 1017 mark-ups were missed under both Condition A and B, meaning no label was assigned to the marked up text from the training set even though the sentence had originally been given a category label. A total of 310 markups were misclassified under Condition A; meaning the label assigned to the text was incorrect. The number of misclassifications increased only slightly to 337 under Condition B, showing that some cases in the case base generalize well to multiple marked up sentences in the training set, capturing not only individual sentences but their meaning. A

reason for the algorithms to miss 16 mark-ups is that the sentences occurring in those markups could not be adequately translated into cases by the text analyzing and case building component in the system.

V. Discussion & Conclusions

This paper presents a prototype classification engine that combines methods from case-based reasoning and natural language processing to learn and identify sensitive information in unstructured text. The methods leverage experience, a key element of human expertise developed within a particular domain and subsequently applied to solve new problems. By storing marked up content as cases in a case base, the classification engine acquires the experience to classify sensitive information based on specified security policies. This experience can then be applied to discover sensitive information in unstructured text. Such an approach avoids the need to explicitly encode the security policies themselves in a machine readable form to automate the process of text classification and markup.

While successes in the use of CBR are encouraging, there has been little work in academia or the industry in applying the CBR approach to problems involving unstructured text. Although CBR techniques can be applied by using feature vectors to represent unstructured text, such methods cannot be applied directly to information-sharing problems. Systems that address information-sharing problems not only must determine the relevance (in the context of type/category) of some information given unstructured text but they also must identify the specific excerpt from the text itself considered to be sensitive.

For this reason, the information-sharing problem can be viewed as a two-fold problem. First, one needs to identify a specific piece of information *within* the text (that which may or may not potentially be shared, e.g., source/method information). Second, for policy implementation purposes one needs to determine the category/classification of the text that contains the aforementioned specific information. For instance, does the entire text fall under a specific category such as NDP Category 8 (Military Intelligence)? To effectively share information, it is not sufficient to only determine that an entire text document contains sensitive content. Depending on who the recipient is, it may actually be necessary to point out the specific sensitive content. The proposed approach in this paper addresses the two-fold information-sharing problem by representing security policies as cases in a case base that are designed to release information from a higher to a lower security domain. Case knowledge can later be applied to quickly identify information within text documents that is considered sensitive with respect to the low security domain and therefore must be removed prior to release.

Preliminary results have shown that the proposed methods successfully detect and label sensitive content in unstructured text. Further work is needed to improve the accuracy by which sentences are compiled into case representations. Also, methods are needed to retrain the classifier based on feedback from reviewers. For example, reviewers need to be able to select incorrect mark-up suggested by the classifier and assign it the correct sensitivity label or make it non-sensitive.

In addition, existing methods must be extended to develop scalable solutions. The current techniques require that each sentence be compared to every case in the case base to assess its sensitivity level. However, the case-base itself grows linearly, as more and more marked up text is fed into the system slowing down the execution of any new markup task. Case-based maintenance ([3],[10]) addresses performance issues typically associated with continued updates of case bases by removing selected cases while preserving case-base competence. Other approaches to consider include hierarchical case-based reasoning [9] that combines both abstract and concrete case representations to solve problems. These approaches could be adapted to build case representations at different levels of granularity to identify sensitivity content in text documents by analyzing first entire documents, then paragraphs, and finally sentences, allowing the text classification systems to quickly discern sensitive from non-sensitive content.

Acknowledgment

Work described in this paper was funded by the Air Force Research Lab (AFRL), Rome, NY under an SBIR Phase I and II grant, contract numbers FA8750-08-C-0110 and FA8750-07-C-0117.

References

[1] J.L. Kolodner (1993). *Case-Based Reasoning*. Morgan Kaufmann.

[2] D.B. Leake (1996). *Case-Based Reasoning: Experiences, Lessons and Future Directions*. AAAI Press/MIT Press.

[3] D.B. Leake, D.C. Wilson (2000). Remembering Why to Remember: Performance-Guidled Case-Base Maintenance, *Proceedings of EWCBR-2K*, Springer-Verlag, Berlin.

[4] D. Gildea & D. Jurafsky (2002). Automatic Labeling of Semantic Roles. *Computational Linguistics*, 28:3, 245-288.

[5] D. Lin (1998a). An Information-Theoretic Definition of Similarity. *Proceedings of the International Conference on Machine Learning*. Madison, Wisconsin.

[6] S. Pradhan, K. Hacioglu, V. Krugler, W. Ward, J. Martin, & D. Jurafsky (2005). Support Vector Learning for Semantic Argument Classification. *Machine Learning*, 60:11-39.

[7] M. Sahami, S. Dumais, D. Heckerman, & E. Horvitz (1998). A Bayesian approach to filtering junk e-mail. AAAI'98, *Workshop on Learning for Text Categorization*.

[8] G. Salton & M.J. McGill (1986). *Introduction to Modern Information Retrieval*. McGraw-Hill.

[9] B. Smyth, P. Cunningham, & M. Keane (2001). Hierachical Case-Based Reasoning: Integrating Case-based and Decompositional Problem-Solving Techniques for Plant-Control Software Design. *IEEE Transactions on Knowledge and Data Engineering*. 13:5, 793-812.

[10] B. Smyth and M. Keane (1995). Remembering to forget: A competence preserving case deletion policy for case-based reasoning systems. In *Proceedings of the Third International Conference on Case-based Reasoning*. Springer Verlag, Berlin.