Ranking Algorithms For Digital Forensic String Search Hits

By

Nicole Beebe and Lishu Liu

From the proceedings of

The Digital Forensic Research Conference

DFRWS 2014 USA

Denver, CO (Aug 3rd - 6th)

Digital Investigation 11 (2014) S124–S132

Contents lists available at ScienceDirect

Digital Investigation

journal homepage: www.elsevier.com/locate/diin

Ranking algorithms for digital forensic string search hits

Nicole Lang Beebe*, Lishu Liu

The University of Texas at San Antonio, Department of Information Systems and Cyber Security, San Antonio, TX, USA

Keywords:
Digital forensics
String search
Ranking
Ranked list
Relevancy

A B S T R A C T

This research proposes eighteen quantifiable characteristics of allocated files, unallocated clusters, and string search hits contained therein, which can be used to relevancy rank string search output. We executed a 36-term query across four disks in a synthetic case ("M57 Patents" from DigitalCorpora.org), which produced over two million search hits across nearly 50,000 allocated files and unallocated clusters. We sampled 21,400 search hits from the case, extracted the proposed feature values, trained binary class (relevant/not-relevant) support vector machine (SVM) models, derived two relevancy ranking functions from the resultant model feature weights, and empirically tested the ranking algorithms. We achieved 81.02% and 85.97% prediction accuracies for the allocated and unallocated models, respectively. Further research is needed to validate these algorithms in a broader set of real-world cases, and/or adapt the algorithms to improve their robustness. Nonetheless, this research provides an important starting point for research into digital forensic search hit relevancy ranking algorithms. We proposed an initial set of relevancy ranking features and obtained very promising empirical results. The ability to achieve rank-ordered list output for search queries in digital forensics, similar to what web browsing and digital library users enjoy, is extremely important for digital forensic practitioners to reduce the analytical burden of text string searching – a valuable analytical technique.

Introduction

Textual evidence is important in many investigations, given common user activities like email, web browsing, word processing, etc. However, even well-formed queries can easily net millions of search hits with very low relevancy rates. As a result, investigators often rely on other search techniques, such as file carving, file hashing, and logical file analysis. However, doing so risks missing important textual evidence, for example when the search hit exists in a file fragment in unallocated space for which the file signature is overwritten.

Search hit ranking algorithms are prevalent in traditional information retrieval (IR) contexts, such as web browsing and digital library searching. However, digital forensic investigators are unable to leverage many of the features of those ranking algorithms for physical-level disk searches, given the vast differences in domain and data characteristics. With non-relevancy rates commonly $\leq$5% amidst millions of search hits, it is important that researchers develop robust search hit ranking algorithms (Beebe and Clark, 2007). Digital forensic investigators need mechanisms to effectively and efficiently leverage string search output to find digital evidence in investigations.

The purpose of this research was to propose and empirically validate relevancy ranking algorithms for digital forensic string search hit output. The paper is outlined as follows. First, we identify and support our proposed ranking features. Next we explain our methodology for empirically deriving and validating the ranking algorithms, including feature operationalization, datasets, and model building. Then we present empirical model testing results, derive linear discriminant ranking functions, and analyze

* Corresponding author. Tel.: +1 210 458 8040; fax: +1 210 458 4732.
E-mail address: nicole.beebe@utsa.edu (N.L. Beebe).

http://dx.doi.org/10.1016/j.diin.2014.05.007
1742-2876/

the results from the perspective of model performance and feature influence on relevancy rank. Finally, we conclude with a discussion of limitations, contributions, and provide concluding remarks.

Proposed features

We propose eighteen (18) features as quantitative indicators of search hit relevancy, which are listed below and discussed thereafter.

- *Recency-Created*: Amount of time passed between allocated file creation and a specified reference point (e.g., time of forensic analysis, specific instance of unauthorized access, etc.)
- *Recency-Modified*: Amount of time passed between allocated file last modification and specified reference point.
- *Recency-Accessed*: Amount of time passed between allocated file last accessed time and specfied reference point.
- *Recency-Average*: Average of recency-created, recency-modified, and recency-accessed to lessen the impact of an anomalous MAC date/time stamp that may occur due non-case related file activity (e.g., virus scanning of file content).
- *Filename-Direct*: Hit exists in a file/path name.
- *Filename-Indirect*: Hit is contained in the content of an allocated file, whose file/path name contains a different search term.
- *User Directory*: Hit is contained in an allocated file found in a non-system directory.
- *High Priority Data Type*: Hit is contained in a high priority data type (see Table 1, however, prioritization may be case specific).
- *Medium Priority Data Type*: Hit is contained in a medium priority data type (see Table 1).
- *Low Priority Data Type*: Hit is contained in a low priority data type (see Table 1).

Table 1
Data type priority classes.

High Priority Data Types (with examples)
Word Processing (doc, docx, odt, rtf, tex, txt, wpd, wps)
Email Related (eml, emlx, ics, mbox, mdb, msf, msg, ost, pab, pbx, pdb, pst, vcf, wdseml)
Spreadsheet (csv, xls, xlsx, xlr)
Presentations (ppt, pptx, key, pps)
Web (htm, html, mht, urlencoded data)
Database (accdb, db, dbf, dbx, sql, sqlite)
Readers (pdf, pages, ps)
Medium Priority Data Types (with examples)
Encoded data[a] (base85, base64)
Compressed data[a] (7z, bzip, bzip2, gz, rar, tar, zip, zipx)
Markup (json, xml)
System (dat, data, ini, lnk, sdf, sys)
File system (ext3, FAT, NTFS)
Logs (evt, evtx, log)
Other (bak, efax, tif, tiff, tmp)
Low Priority Data Types: All other types not listed above

[a] Encoded or compressed data that could not be decoded or uncompressed (i.e. fragment in unallocated space statistically classified as the referenced type).

- *Search Term TF-IDF*: Number of times search term occurs in the corpus (i.e. entire physical disk, if physical level search), moderated by inverse document frequency of the search term across the corpus (see Eqs. (1) and (2)).
- *Object-level hit frequency*: Number of times search term occurs in an allocated file or unallocated cluster.
- *Cosine similarity*: Traditional cosine similarity measure between the vectors representing the search query and the object containing the search hit (allocated file or unallocated cluster).
- *Search hit adjacency*: Byte-level logical offset between adjacent hits (next nearest neighbor) within an allocated file or unallocated cluster.
- *Search term object offset*: Byte distance between the start of the allocated file or unallocated cluster and the logical level offset of the search hit.
- *Proportion of search terms in object*: Number of different search terms that appear in the allocated file or unallocated cluster, divided by the total number of search terms in the query.
- *Search term length*: Byte length of search term.
- *Search term priority*: User ranked priority of search term, relative to the other search terms.

The *recency* features pertaining to MAC date/time stamps are proposed because of the known importance of timeline analysis to digital forensic investigations (Carrier and Spafford, 2003; Buchholz and Spafford, 2004). System and file data/time activity relative to some reference point, such as a specific instance of unauthorized access, can be an important indicator of search hit relevancy. In our experience, a common search heuristic employed by practitioners is to sort search hits from allocated files based on date/time stamps.

The two features pertaining to *filename* are analogous to Google's various ranking features that pertain to keyword existence in important metadata (e.g., the hostname, path segment in the URL, title tag, description tag, keyword tag, or various heading tags) (Su et al., 2010).

The *user directory* and *data type priority* features rely on the frequent probability that user created data contained in certain file types and/or stored in user directories are more relevant to an investigation than non-user data. Certainly, this is not always the case. System related data and/or data located in system directories may be more relevant in some cases (e.g., intrusion investigations). However, this feature can be operationalized differently in different cases, and future research may explore alternative relevancy ranking models for various scenarios. A user of the search hit ranking algorithm might specify at the outset whether user or system data and/or directories are more important, based on known facts about the case, or common tactics, techniques, and procedures for the case type being investigated. The investigator may also uniquely assign data/file types to specific priority classes (or priority scales, eventually). Data type priority class assignment for our experiments was as is specified in Table 1.

Our data type priority class assignment was based on a number of factors, including: 1) knowledge about the investigative goals for the synthetic case and the anticipated type of evidential artifacts of interest, 2) prioritization of user related file types relative to system related artifacts, and

3) prioritization of data that is not in a compressed, encoded, or encrypted state when search strings are found. Note, our search decompressed data prior to string searching where possible (e.g., common email container types and common, decompressible, in-tact zip files).

Search term TF-IDF, *object level hit frequency*, and *cosine similarity* are classic IR measures for measuring query-result similarity. The greater the quantitative similarity, the higher the hit and/or document is ranked. *Search term TF-IDF* identifies high frequency terms, but penalizes those that appear in many documents across the corpus, and thus carry little information or discriminating value.

Object level hit frequency reflects the frequency of the search expression within a specific allocated file or unallocated cluster, whereas our TF-IDF measure reflects corpus level TF. Accordingly, our TF-IDF operationalization provides a corpus level measure of the search term's importance, while the TF operationalization provides a document level measure of the search term's importance (referred to as 'keyword density' in web retrieval).

The last of our traditional IR measures proposed is *cosine similarity*. It is the classic cosine of the angle θ between a document d_j and a query q, represented as t-dimensional vectors, where $t \in \{\text{query terms}\}$. Smaller angles mean greater similarity between the responsive object and the query.

Search hit adjacency and *search term object offset* are analogous to existing web retrieval ranking features. *Adjacency* is akin to Google's 'proximity' measure—the distance between search hits in a multi-hit, responsive object (Brin and Page, 1998). *Object offset* is akin to Google's feature that ranks responsive documents higher when the keyword exists in the first 50–100 words in the mark-up text of a document. The *adjacency* and *object offset* features are byte-level features in our context—byte-offset between search hits within a document and byte-offset from the start of the allocated file or unallocated cluster.

Proportion of search terms in the object is motivated by the theory that the more search query terms a responsive object contains, the more likely it is relevant to investigative goals.

We proposed *search term length* based on past research that has shown that word length is a useful measure in search hit ranking algorithms (Lynch et al., 2004). This finding is in line with keyword list heuristics commonly used in practice, in our experience.

Last, *search term priority* is proposed since it is reasonable to believe that allocated files and unallocated clusters responsive to higher priority search terms (as specified by the user) are likely to be more relevant than those responsive to lower priority search terms. This feature is also proposed based on the common information retrieval practice of using query element ordering as a proxy for search term priority.

Methodology

Feature extraction and operationalization

We used a number of tools for feature extraction, including: (1) `EnCase™` as the string search tool; (2) `The Sleuth Kit` binaries to extract additional features of interest from allocated files and unallocated clusters containing hits; (3) `Sceadan` (Beebe et al., 2013) to data type classify unallocated clusters; and (4) purpose-built `python` scripts for feature normalization procedures and vector formation (to transform the data into the format needed for experimentation).

We measured and operationalized each feature as follows:

- *Recency features*: data extracted from the $STANDARD_INFORMATION attribute from $MFT records; difference between date/time stamp and a reference point (specified as date of forensic analysis in this case, but may differ in other cases); normalized by maximum time difference in corpus (difference between oldest date/time stamp and reference point); continuous feature with range $f = \{0 \ldots 1\}$, with lower values being closer to reference point.
- *Filename-Direct*: simple pattern match operation for the hit's search expression in the file's path/filename; binary feature with $f = \{0|1\}$.
- *Filename-Indirect*: simple pattern match operation for other search expressions in the file's path/filename; binary feature with $f = \{0|1\}$.
- *User directory*: specified standard Windows system directories and defined user directories as all non-system directories; binary feature with $f = \{0|1\}$.
- *Data type priority*: specified high-medium-low data type tables; used file signatures of allocated files for type identification; used `Sceadan`, a naïve statistical data type classifier, for data type classification of unallocated clusters; binary feature with $f = \{0|1\}$ for each priority level.
- *TF-IDF*: used normalized, logarithmic, corpus level term frequency (Lo et al., 2005) (see Eq. (1)), moderated by inverse document frequency (see Eq. (2)); continuous feature with range $f = \{0 \ldots 1\}$.

$$\mathrm{TF}_{\mathrm{norm}} = -\log\left(\frac{\mathrm{TF}}{v}\right), \tag{1}$$

where TF = count in corpus; v = total tokens in corpus; token = alphanumeric string ≥ 2 bytes in length

$$idf_k = \log\left(\frac{\mathrm{NDoc}}{D_k}\right), \tag{2}$$

where NDoc = total no. of objects in corpus; D_k = no. of objects containing term (k); objects = allocated files and unallocated clusters

- *Cosine similarity*: measured by the traditional IR cosine similarity measure between the document and the query (Baeza-Yates and Ribeiro-Neto, 1999) (see Eq. (3)); normalized by the highest cosine similarity measure returned; continuous feature with range $f = \{0 \ldots 1\}$. Note, our cosine similarity measures were never negative, because both vectors as defined are positive.

$$\mathrm{sim}(d_j, q) = \frac{\vec{d}_j \bullet \vec{q}}{\left|\vec{d}_j\right| \times |\vec{q}|} \tag{3}$$

- *Object-level hit frequency*: measured by the term frequency (TF) of the search expression in the allocated file or unallocated cluster; normalized by the highest TF returned; continuous feature with range $f = \{0 \ldots 1\}$.
- *Search hit adjacency*: distance (in bytes) between search expression and the most proximally located search hit for a different search expression; measured via logical file offset in allocated files to account for fragmentation effects on distance; normalized the largest adjacency distance returned; continuous feature with range $f = \{0 \ldots 1\}$.
- *Proportion of search terms in object*: total number of search expressions that exist in the allocated file or unallocated cluster; normalized by the maximum number of search expressions per object returned; continuous feature with range $f = \{0 \ldots 1\}$.
- *Search term length*: search expression's length in bytes; normalized by maximum length of any search expressions; continuous feature with range $f = \{0 \ldots 1\}$.
- *Search term priority*: measured by rank-ordering of the search expressions by the user specified priority order; normalized by the highest numeric rank returned; continuous feature with range $f = \{0 \ldots 1\}$.
- *Search term object offset*: measured by file offset of the search expression from the start of the allocated file or unallocated cluster; normalized by largest search term object offset value returned; continuous feature with range $f = \{0 \ldots 1\}$.

Dataset and sampling procedures

We used a single synthetic case, involving four users' workstations—specifically the 'police seizure' images from the M57 Patents dataset (Garfinkel, 2009; Garfinkel et al., 2009). The data set was constructed by researchers, according to a scripted scenario, in a manner that amassed evidentiary artifacts amongst 'normal user activity' to the maximum extent possible in a synthetic case. Significant user activity occurred daily for 19 days, before the 'police seizure' images were created. Accordingly, while the data set is relatively small with regard to disk size and organization size, it is a high-quality, publicly available data set.

We conducted a 36-term search query, using prioritized search terms shown in Table 2. The search query was formed by compiling search strings recommended by several skilled digital forensic investigators, after having received basic information about the case and the investigative goals. We specifically instructed the investigators to identify search strings *without regard* to the potential for false positives, for the following reasons: 1) to mitigate the impact of poorly formed search queries, and 2) to enable investigators to search for terms that are of interest, but traditionally ill-advised to use.

The search returned 2,640,681 search hits located in 46,884 allocated files and unallocated clusters. Of these, 4.24% (112,020 hits) were relevant to investigative objectives. Search hit relevancy was determined by a single human subject with formal digital forensics training and knowledge of the scenario's investigative objectives, but prior to the execution of any experiments to avoid biasing the empirical results. Because of the sheer volume of the relevancy determination task, the analyst individually reviewed all search hits for search terms other than the "M57" search term – the term that produced the largest number of hits, as would be expected given the string is the name of the business. The analyst then assigned relevancy to the M57 search hits based on the object in which it appeared, deeming the hit relevant if found in an object with other relevant hits, or non-relevant if found in an object without any relevant hits from the prior analysis. This heuristic is sub-optimal, but was needed since even averaging one second per search hit, the relevancy determination task would have taken 18 weeks, full-time – further punctuating the need for digital forensic string search hit ranking algorithms.

Table 2
Search terms and prioritization.

Priority:	Search term	Priority:	Search term
1:	mcgoo01	19:	Terry Johnson
2:	johnson01	20:	Jo Smith
3:	smith01	21:	Pat McGoo
4:	brown01	22:	Charlie Brown
5:	VNC.exe	23:	terry@m57.biz
6:	netcat	24:	jo@m57.biz
7:	Project2400	25:	pat@m57.biz
8:	swexpert.com	26:	charlie@m57.biz
9:	steg	27:	cancer
10:	VNC	28:	@m57.biz
11:	nc	29:	immortality
12:	prior art	30:	prostate
13:	Nitroba	31:	Epithelial
14:	Patent	32:	US000
15:	ftp	33:	M57
16:	virus	34:	CEO
17:	jaime	35:	IT
18:	money	36:	PR

For experimentation, we selected all of the relevant hits from each disk and a random sample of non-relevant hits equal in number to the number of relevant hits selected. This ensured our sample was balanced, considering the ratio of relevant to non-relevant search hits, and ensured our prediction results were not biased toward non-relevant prediction simply because of class imbalance. Otherwise, search hit relevancy predicted class would be biased toward non-relevant, simply because of probabilistic norms. Given the stark relevant:non-relevant class imbalance at play in this domain, such a bias would lead to good overall prediction accuracy rates, but a high false negative rate, which is contrary to salient search objectives. This sample selection procedure was followed for both allocated and unallocated hits for the *Charlie* disk, however, only allocated hits were included for the other disks, due to insufficient sample sizes for unallocated hits on the other disks. Table 3 summarizes the sample sizes from each disk.

Model building

We utilized support vector machines (SVMs) (Vapnik, 1998), a popular supervised machine learning technique for data classification. In our case, we sought to build a binary-class SVM, where the two classes are *relevant* and

Table 3
Sample sizes (no. of hits).

Disk	Alloc. Relevant	Alloc. Non-rel.	Unalloc relevant	Unalloc Non-rel.
Charlie	1900	1900	900	900
Jo	3300	3300	0	0
Pat	2800	2800	0	0
Terry	1800	1800	0	0
ALL	**9800**	**9800**	**900**	**900**

non-relevant. The binary SVM not only has the ability to determine relevancy of a document based on its predicted class, but it also allows us to build a linear discriminant ranking function. Since the binary-class feature weights produced by the SVM model can be used in as coefficients in a linear discriminant function, we can calculate a relevancy *score* for search hits. This makes it possible to quickly rank ordered list display of digital forensic search hits.

After some initial experimentation, we found the linear kernel SVM performed well enough to use it over the slower radial basis function (RBF) SVM. We also ran logistic regression models using `liblinear's` (Fan et al., 2008) L1 and L2 regularized logistic regression functions. The prediction accuracies were comparable to, but did not exceed the linear model results. The relative feature weights were also comparable. As a result, we elected to use and present the linear model results in this research.

We used `liblinear`, using both the L2 regularized L2 loss function, primal solver, and the L2 regularized L1 loss function, dual solver, both with default termination criterion $\varepsilon = 0.01$. For the regularization parameter C (the penalty for each misclassification during model training—higher C values build a more accurate model for the data, but too high results in over-fitting), we conducted five-fold cross-validation experiments to empirically determine the optimal C-value using `LIBSVM's` (Chang and Lin, 2011) `grid.py` function. Final parameter values are shown in the Findings section (see Table 3). A full explanation of the functions, solvers, and parameters can be found in Fan et al. (2005).

We used a 60%:40% train:test ratio, randomly selecting the training set without replacement, for model building and testing. We selected this train:test ratio to minimize model over-fitting that would tend to occur with a larger train:test ratio. We did not use cross-validation procedures for model building and testing, because our sample size was of sufficiently sized that cross-validation procedures were not necessary. (Cross-validation permits scientists to use all of the data for model training, while minimizing the effects of over-fitting. However, results are most robust and generalizable when less of the data is used in the model building process.)

We built separate ranking models for search hits in allocated files vs. search hits in unallocated clusters, since not all features are applicable to unallocated space (e.g., date/time stamp features).

Table 4
Model testing on 40% random hold-out sample.

	C value Solver	Predict. Accuracy	Train time	Predict Time[a]
Alloc. Hits	0.5 Dual	80.83%	0.006s	0.026s
	0.5 Primal	81.02%	0.077s	0.022s
Unalloc. Hits	32 Dual	85.97%	0.017s	0.013s
	32 Primal	83.75%	0.018s	0.003s

[a] Using 16 core server with 2 GHz processors and 32 GB of RAM.

Table 5
Confusion matrix – allocated model.

True/Predict	Not-relevant	Relevant
Not-Relevant	2949	971 (Type I)
Relevant	517 (Type II)	3403

Findings

Model testing

We tested the models separately on the 40% randomly selected hold-out sub-sample. Table 4 shows the relevancy prediction results, which were very comparable to the five-fold cross-validation results observed during the parameter optimization selection procedures. The primal solver outperformed the dual solver on the allocated model, but the opposite was true for the unallocated model. The unallocated hits model slightly outperformed the allocated hits model (85.97% prediction accuracy, compared to 81.02% prediction accuracy). With a 60%:40% train:test ratio without cross-validation to minimize over-fitting risks, the prediction accuracies are promising.

Our Type I (false positive) error rate exceeded our Type II error (false negative) rate in both models, as shown in Tables 5 and 6. This is advantageous, since in this context investigators prefer to minimize false negatives, so as to avoid missing relevant evidence. Analysis of the false negatives for both models disclosed high percentages of hits near the relevancy threshold. Accordingly, the relevancy threshold could be lowered in practice to further minimize false negatives if desired.

Last, we analyzed our ranking algorithm performance by sorting the test sample hits by predicted relevancy score and measuring recall, precision, and average precision at quartile increments. The prediction accuracies reported in Table 4 indicate successful binary classification. However, Tables 7 and 8 suggest the continuous value ranking scores for each hit will translate well into rank-ordered list output.

Note, average precision is a score that considers the order in which relevant and non-relevant hits are presented in a rank-ordered list. It is a good measure of ranking

Table 6
Confusion matrix – unallocated model.

True/Predict	Not-relevant	Relevant
Not-Relevant	285	75 (Type I)
Relevant	26 (Type II)	334

Table 7
Performance – allocated model.

No. Hits reviewed	Recall	Precision	Average precision
25%	0.42	0.84	0.37
50%	0.80	0.80	0.68
75%	0.96	0.64	0.80
100%	1.00	0.50	0.82

algorithm quality, as it only penalizes low precision output when the ranking algorithm fails to rank the relevant hits higher than the non-relevant hits (see Eq. (4)).

$$\text{Average Precision (AvgP)} = \frac{\sum_{r=1}^{N} P(r) \times \text{rel}(r)}{R} \quad (4)$$

where r = rank

N = number hits retrieved
$\text{rel}(r) = 0$ or 1 (relevancy of hit)
$P(r)$ = total precision up to this point
R = Total number of relevant hits

Linear discriminant function coefficients

The relevancy ranking function used to calculate continuous value relevancy scores, as opposed to a predicted relevancy class, is a simple linear discriminant function applying the binary class SVM feature weights as feature (f_n) coefficients (w_n). The relevancy rank of a search hit is then the summation of weighted feature values, as show in Eq. (5).

$$R_{\text{hit}} = \sum_{n=1}^{18} w_n f_n \quad (5)$$

Table 9 provides the feature weights derived from our linear kernel binary class SVM models. These weights are then used as the weights in Eq. (5).

Discussion

Model performance

The empirical results support the proposition that ranking algorithms for digital forensic string search hit output are feasible and worthwhile. The relevancy ranking algorithm for allocated hits performed well with 81.02% prediction accuracy on the 40% randomly selected, hold-out test sample. The unallocated relevancy ranking algorithm also performed well with 85.97% prediction accuracy on the 40% test sample, although this performance involved only one disk. Having less evidentiary variety than the allocated model during training and testing, may explain the superior performance of the unallocated model in our experiments.

Table 8
Performance – unallocated model.

No. Hits reviewed	Recall	Precision	Average precision
25%	0.46	0.92	0.43
50%	0.86	0.86	0.79
75%	1.00	0.66	0.90
100%	1.00	0.50	0.90

Table 9
Binary class SVM feature weights.

Feature	Alloc. Model	Unalloc. Model
Recency-Created	−0.46644558	N/A
Recency-Modified	0.18766033	N/A
Recency-Accessed	1.00085313	N/A
Recency-Average	0.24545862	N/A
Filename-Direct	−0.79529512	N/A
Filename-Indirect	2.75526926	N/A
User Directory	−1.93197353	N/A
High Priority Data Type	0.35261256	−0.07062238
Medium Priority Data Type	0.28760329	−0.11257353
Low Priority Data Type	0.26579067	−0.08896686
TF-IDF of search term	3.18077517	−2.06304540
Cosine similarity	−0.13591579	−0.25250011
Object-level hit frequency	0.30010899	1.50116329
Search hit adjacency	−0.27918942	0.42475082
Proportion–search terms in object	2.05643916	0.84795958
Search term length	4.11034658	2.81213633
Search term priority	−3.45112479	−3.55140039
Search term object offset	−0.61271427	−0.20266161

Feature significance

Analysis of the binary SVM feature weights provides important insight into the nature and relative importance of each feature in predicting search hit relevancy. The features with higher absolute magnitude provide more 'input' to the relevancy prediction function than those with lower absolute magnitudes. The sign of the feature weight provides insight into whether the feature helps predict a search hit *is* (positive weight) or *is not* (negative weight) relevant to investigative goals.

Figs. 1 and 2 highlight the relatively important features in our models. We clearly observe the two most influential features in both models are *search term length* and *search term priority*. It may at first glance appear counterintuitive that the feature weight for *search term priority* is negatively signed, however, in our operationalization, the search expression with `rank = 1` is higher priority than those with `rank>1`. So, the negative sign indicates that the higher the numeric rank value of a hit's search expression, the less relevant it is likely to be, which matches our expectations.

Another important feature in both models is the *proportion of search terms in an object*. This means that allocated files and unallocated clusters containing multiple search expressions are likely to be more relevant than those containing fewer search expressions.

Features important to the allocated model, which are not applicable to the unallocated model, include: *filename-direct, filename-indirect, recency-accessed,* and *user directory*. The results suggest that if the hit is contained in a file whose filename or pathname contains another search expression, it is likely to be more relevant than a hit that does not contain a search expression in its filename or pathname.

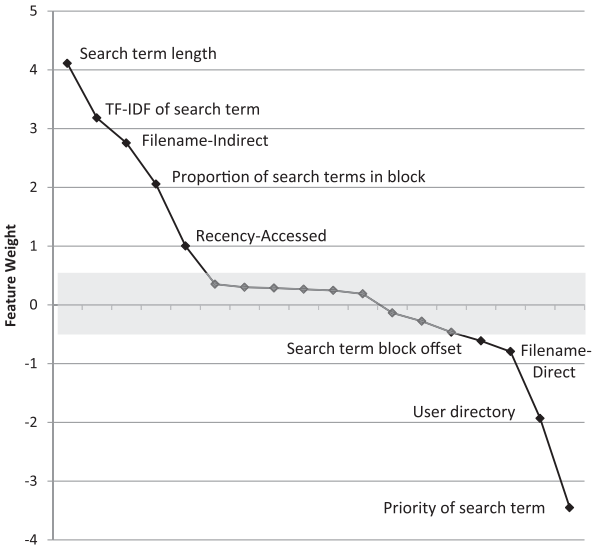

Fig. 1. Allocated model feature weights.

Filename-direct and *user directory* emerge as significant discriminators, but exhibit opposite sign than would be expected. Further research is needed to determine if the unexpected sign-reversal can be explained by correlations with other features, or case specific characteristics of the synthetic dataset.

The TF-IDF of the search expression proved to be an important feature in both models, however, it was positively signed in the allocated model and negatively signed in the unallocated model. The positive sign of the TF-IDF feature weight in the allocated model matches our expectation. The more times the search expression exists in the corpus, moderated by the inverse document frequency statistic, the more relevant associated hits are likely to be.

The negative sign of the TF-IDF feature weight in the unallocated model, however, is counter-intuitive. It is possible that the sign-reversal is a misleading artifact when considered alone, and really its weighted impact is in concert with the other features measured and used in the overall equation. This may be the case, considering its correlated *object-level hit frequency* feature emerges as a prominent feature in the unallocated model.

Last, *search term object offset* is an important feature in the allocated ranking model. Its sign is negative, because as the search expression exists later in the logical document, its feature value approaches one $(f \rightarrow 1)$. This means, the further away from the start of the file the expression is, the less relevant the hit is likely to be. This is somewhat intuitive, as more salient, information bearing content often exists at the beginning of the document. This is not always true, however, especially considering file types with long headers. This may explain why this feature emerges as important to the relevancy prediction, but not as significant as other features discussed thus far.

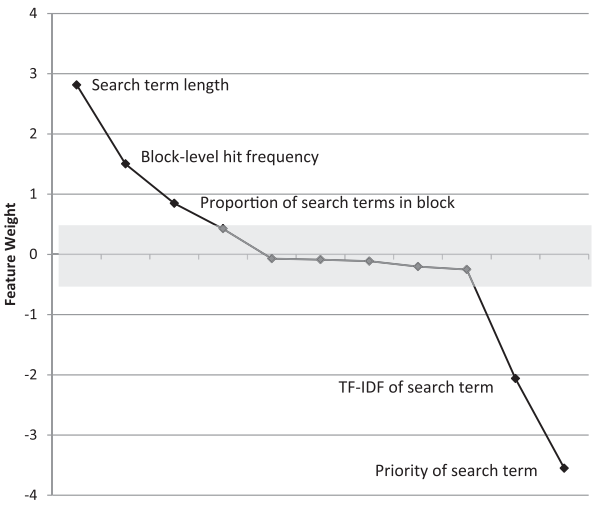

Fig. 2. Unallocated model feature weights.

Limitations

While the models performed well, the validity and extensibility of the models remain in question, since the models were validated on search hits responsive to the same search query within the same case. The findings are somewhat promising, however, given the diversity of data across systems in the M57-Patents case. The simulated users were engaged in different computing activities, both with respect to their job function, as well as their illegal activity, or role therein.

- Charlie was a patent researcher involved in corporate espionage, extortion, and document exfiltration.
- Jo was also a patents researcher, but was involved in an entirely different crime—contraband materials (pictures and videos).
- Terry was an IT administrator, and was siphoning money and equipment from the company to pay gambling debts.
- Pat was the CEO/Founder and was not involved in any criminal activity.

The results are also limited by the fact that single synthetic data set was used in the primary empirical analysis. Further research is needed to validate these algorithms in a broader set of synthetic *and* real-world cases, and/or adapt the algorithms to improve their robustness.

Contributions

Related work is extremely limited (Beebe and Clark, 2007; Lee, 2008). Past research is non-empirical and limited to theoretical models only, with limited feature identification (Lee, 2008). We proposed 18 quantifiable measures of search hit relevancy and empirically determined the importance of each. We empirically derived and tested promising ranking algorithms for digital forensic string searching. We have instantiated our algorithms in a professionally developed, open source tool `Sifter`. The tool ingests digital forensic image files of popular formats

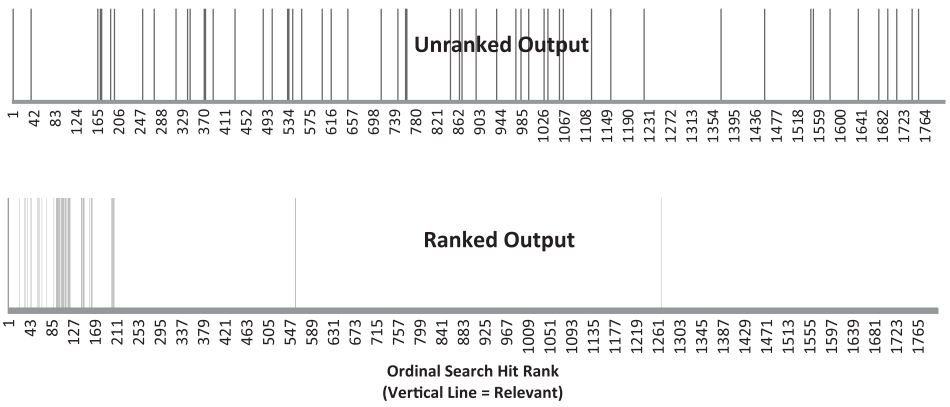

Fig. 3. Unranked vs. ranked output comparison.

(dd, .E01, .AFF), indexes the entire image at the physical level, and enables standard Boolean querying by the user. The resultant output is provided to the user in a traditional "table view" of string search hits, but for the first time, hits are now ranked by the relevancy ranking algorithms derived under this research. Use of the tool on a different synthetic data set provides anecdotal evidence that the ranking algorithm may be extensible to other data sets, but further research is needed to formally validate and test the generalizability of the algorithm (see Fig. 3). The vertical lines represent relevant hits, and the *x*-axis is the ordinal position of each hit in the list-ordered output (ranked in one case, and unranked in the other case). Clearly, the rank-ordered list created using our algorithms provides marked improvement, moving relevant hits closer to the "top" of the search hit output list, so that the analyst can find relevant evidence much more quickly.

Conclusion

Text string searching in digital forensics has decreased in popularity over the past decade, because of the overwhelming analytical burden associated with it. However, textual artifacts are important to many digital forensic investigations. Sometimes they can be located and recovered through alternate analytical means, such as hash analysis, file carving, and logical file analysis, as well as advanced string search approaches like file print analysis (Shields et al., 2011). However, a fundamental question remains—-why don't digital forensic investigators enjoy the same basic rank-ordered list output that we all enjoy with web search engines and digital libraries?

This research closes that gap by proposing 18 quantitative features for search hit ranking and empirically deriving and testing two ranking algorithms—one for search hits in allocated files and one for search hits in unallocated space. We show significant improvements in IR effectiveness with rank-ordered list output. The results are very promising, suggesting ranking algorithms can be developed and implemented quickly and easily. Further research is needed to test the robustness of our algorithms, explore the utility of case type specific ranking algorithms, identify and test new ranking features, and fine tune our feature weights with a greater number and variety of test cases.

Acknowledgments

This publication was developed under work supported by the Naval Postgraduate School Assistance Agreement No. N00244-11-1-0011, awarded by the Naval Supply Systems Command (NAVSUP) Fleet Logistics Center San Diego (NAVSUP FLC San Diego). It has not been formally reviewed by NPS. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the NPS or NAVSUP FLC San Diego. The NPS and NAVSUP FLC San Diego do not endorse any products or commercial services mentioned in this publication.

References

Baeza-Yates R, Ribeiro-Neto B. In: Modern information retrieval, vol. 463. New York: ACM press; 1999.
Beebe NL, Clark JG. Digital forensic text string searching: improving information retrieval effectiveness by thematically clustering search results. Digit Investig 2007;4(S1):49–54.
Beebe N, Maddox L, Liu L, Sun M. Sceadan: using concatenated N-Gram vectors for improved file and data type classification. Inf Forensics Security, IEEE Trans 2013;8(9):1519–30.
Brin S, Page L. The anatomy of a large-scale hypertextual Web search engine. Comput Networks ISDN Syst 1998;30(1):107–17.
Buchholz F, Spafford E. On the role of file system metadata in digital forensics. Digit Investig 2004;1(4):298–309.
Carrier B, Spafford EH. Getting physical with the digital investigation process. Int J Digit Evid 2003;2(2):1–20.
Chang CC, Lin CJ. LIBSVM: a library for support vector machines. ACM Trans Intell Syst Technol (TIST) 2011;2(3):27.
Fan RE, Chen PH, Lin CJ. Working set selection using second order information for training support vector machines. J Mach Learn Res 2005;6:1889–918.
Fan RE, Chang KW, Hsieh CJ, Wang XR, Lin CJ. LIBLINEAR: a library for large linear classification. J Mach Learn Res 2008;9:1871–4.
Garfinkel S, M57-Patents Scenario, 2009. http://digitalcorpora.org/corpora/scenarios/m57-patents-scenario.
Garfinkel S, Farrell P, Roussev V, Dinolt G. Bringing science to digital forensics with standardized forensic corpora. Digit Investig 2009;6:S2–11.
Lee J. Proposal for efficient searching and presentation in digital forensics. In: Availability, Reliability and Security, 2008. ARES 08; 2008 [Third International Conference on: IEEE].
Lo RT-W, He B, Ounis I. Automatically building a stopword list for an information retrieval system. J Digit Inf Mgmt; 2005 [Special

Issue on the 5th Dutch–Belgian Information Retrieval Workshop (DIR)].

Lynch P, Luan X, Prettyman M, Mericle L, Borkmann E, Schlaifer J. An evaluation of new and old similarity ranking algorithms. In: Information technology: Coding and Computing, 2004; 2004 [Proceedings. ITCC 2004. International Conference on: IEEE].

Shields C, Frieder O, Maloof M. A system for the proactive, continuous, and efficient collection of digital forensic evidence. Digit investig 2011;8:S3–13.

Su A-J, Hu YC, Kuzmanovic A, Koh C-K. How to improve your Google ranking: myths and reality. In: Web Intelligence and Intelligent Agent Technology (WI-IAT); 2010 [2010 IEEE/WIC/ACM International Conference on: IEEE].

Vapnik VN. In: Haykin S, editor. Statistical Learning Theory. Adaptive and learning systems for signal processing, communications, and control; 1998.