Chung-Kwei: a Pattern-discovery-based System for the Automatic Identification of Unsolicited E-mail Messages (SPAM)

Isidore Rigoutsos and Tien Huynh

Bioinformatics and Pattern Discovery Group
IBM Thomas J Watson Research Center
Yorktown Heights, NY 10598, USA
rigoutso@us.ibm.com and gdesktop@us.ibm.com

Abstract. In this paper, we present Chung-Kwei[1], a system for the analysis of electronic messages and the automatic identification of unsolicited email messages (=SPAM). The method uses pattern-discovery as its underlying tool and is another instance of a generic approach that has been the basis of previously successful solutions developed by our group to tackle problems in computational biology such as gene finding and protein annotation. Chung-Kwei can be trained very quickly; as new examples of SPAM become available, the system can re-train itself without interrupting the classification of incoming e-mail. We trained Chung-Kwei on a repository of 87,000 messages, then tested it with a very large collection of 88,000 pieces of SPAM and WHITE email: the current prototype achieved a sensitivity of 96.56% whereas the false positive rate was 0.066%, or one-in-six-thousand. In terms of speed, we are currently capable of classifying 214 messages/second, on a 2.2 GHz Intel-Pentium platform. The Chung-Kwei system is part of SpamGuru, a collaborative antispam filtering solution that is currently under development at IBM Research.

1. Introduction

In recent years, electronic mail users around the world have noticed that an ever increasing amount of unsolicited email reaches their mailboxes. Following the surge in the amount of circulating SPAM email, a number of methods have been proposed to address the problem. Some of these include the use of white/black-lists, bulk-email detection, various forms of filtering, and some combination of the above [1,2,3,4].

[1] Chung-Kwei is a significant Feng-Shui figure that is usually shown carrying a bat – a symbol of prosperity and longevity -- and holding a sword behind him. His fierce look makes him ideal as a potent figure for protection. Chung-Kwei is meant to benefit scholars and those who are in businesses involving expensive goods that need to be protected.

Chung-Kwei, which we present below, belongs to the category of filtering schemes. It capitalizes on our earlier pattern discovery work on problems from computational biology that included protein annotation [5], gene finding [6], etc. Chung-Kwei is part of SpamGuru, a collaborative anti-spam filtering solution that is currently under development at IBM Research. It makes use of the Teiresias pattern discovery algorithm first presented in 1998 [9,10]. The Teiresias algorithm has been used to effectively address a very wide spectrum of problems from the life sciences [11,12,13,14,15] and computer security related activities [7,8].

Of the recently proposed methods for recognizing SPAM email, the one that is most similar to Chung-Kwei in spirit is the EMT-Email Mining Toolkit [16]. However, the underpinnings of our effort are orthogonal to EMT in that we operate in a structured pattern-discovery framework, our method is combinatorial (and not statistical) in nature, and more importantly, our method does not seek to identify 'abnormal' behavior.

2. The Method

2.1 The Method: "key-idea"

The idea underlying our method can be summarized as follows: given a collection of SPAM messages, run Teiresias to discover patterns that appear two or more times in this collection (the instances can appear within messages as well as across messages in the collection), then process incoming email messages to see if they match any of the collected patterns: the more patterns that are matched by an email message the more likely it is that the message is bona fide SPAM. As mentioned above, we have used this basic approach – an instance of the "guilty-by-association" methodology which has been very popular in computational biology research for more than 20 years – in a number of life science [5,6] and computer security applications [7,8][2].

Two novel components characterize our work. The first novelty is that we effectively substitute the original knowledge base with an "equivalent" collection of patterns that capture the same amount of information; then, instead of searching the knowledge base for (possibly distorted) instances of a query message, we carry out the reverse operation and actually search the query message for instances of patterns that we have derived from the knowledge base. In other words, we use the discovered patterns as a "SPAM-vocabulary" of sorts which we then use to determine whether a given email message is well-formed with respect to this SPAM-vocabulary at which point we flag the message as SPAM.

[2] In the case discussed in this paper, the "guilty by association" approach operates on the general principal that if a given segment of one email message has a particular property associated with it, then all email messages having that same segment (or some variation of it) also have that property. The "guilty by association" approach is equally applicable to both SPAM and WHITE email messages.

The second novelty is that the representation of the original knowledge with the help of patterns is redundant in that a given location of the knowledge-base will generally participate in more than one pattern: the beneficial impact of this characteristic is the strengthening of the signal-to-noise ratio one obtains during the decision making stage.

For this approach to be successful, the knowledge-base from which to derive the patterns should be a representative sample of the underlying space and, ideally be large and comprehensive. Let us illustrate this last statement with an example that has been adapted from one of our earlier publications [12] – in an effort to be as general as possible and to show how our method is applicable to any kind of input, we will assume that the message to be processed contains no spaces. If the knowledge base contains two messages that read like:

<ahref=”http://getyourmedicationshere.com”>thisisoneexampleofwhatweusuallybeginwith</a>
<ahref=”http://getyourmedicationshere.com”>thisisoneexampleofwhatweusuallybeginwith</a>

then there will be only one discovered pattern and it will coincide with one of these messages, in its entirety. If now the knowledge base was augmented through the addition of

<ahref=”http://wehavethebestmedOcationshere.com”>firstonlyafewmoreexamplestricklein</a>

the following additional patterns would emerge and become part of our SPAM-vocabulary: “med.cationshere.com”>”, “example” and “<ahref=”http://”. As the number of messages in the knowledge-base increases more, potentially redundant patterns, will be discovered.

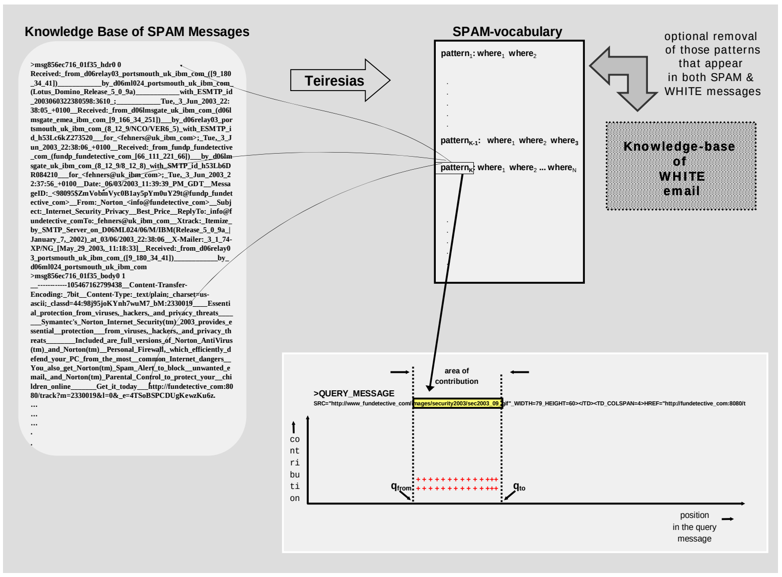
From this simple example, one can see a couple of important emergent properties of our method. Our method does not require the presence of an exact or even near-exact copy of an incoming message in the SPAM knowledge-base before it is able to identify the message as such. A diverse knowledge-base will give rise to a large number of patterns, generally of variable-length. Any incoming message that matches a large number of those patterns will be flagged as SPAM even if the new message shares only parts of it with SPAM messages that have been seen previously[3]. In other words, our method has a built-in ability to extrapolate, to a certain extent. Also, unlike traditional tuple-based approaches such as “k-grams” the patterns that are discovered and used by our method do not have to be specific to the point that each of them can act as a predicate: it is the combined effect of every pattern’s whisper that makes the difference.

2.2 The Method: description

In Figure 1, we show a graphical representation of our method. Two basic stages can be distinguished here. First, we run Teiresias on the knowledge-base of SPAM messages to generate a collection of patterns that cover it as completely as possible. During this operation, we treat each SPAM message as one long line, by removing all intervening carriage-returns.

[3] A related question here is the possibility of “false positives,” i.e. of marking regular email as SPAM. This topic is discussed later in this paper.

This first step takes place off-line, and upon termination the system is ready to classify incoming messages. If a repository of WHITE email is also available, one may wish to remove from the SPAM-vocabulary those patterns that are also encountered in WHITE email. Although not necessary, this optional step is likely to reduce the rate of false positives, if the latter happens to be non-zero; but, for some types of SPAM it may end-up producing the opposite effect and adversely affect the ability to recognize SPAM. Clearly, it is possible to generate separate pattern collections for email-headers and email-bodies. For all of our experiments, we derived patterns only from the bodies of the messages in the SPAM knowledge-base; even though we did not make use of the arguably useful information contained in email headers, we were nonetheless able to achieve impressive performance (see Results section), thus lending further support to the promise of our method.


Knowledge Base of SPAM Messages
Teiresias
SPAM-vocabulary
pattern$_1$: where$_1$ where$_2$
pattern$_{K-1}$: where$_1$ where$_2$ where$_3$
pattern$_K$: where$_1$ where$_2$ ... where$_N$
optional removal of those patterns that appear in both SPAM & WHITE messages
Knowledge-base of WHITE email
area of contribution
contribution
q_{from}
q_{to}
position in the query message

```
>msg856ec716_01f35_hdr0 0
Received:_from_d06relay03_portsmouth_uk_ibm_com_([9_180
_34_41])______________by_d06ml024_portsmouth_uk_ibm_com_
(Lotus_Domino_Release_5_0_9a)_____________with_ESMTP_id
_2003060322380598:3610_;_____________Tue,_3_Jun_2003_22:
38:05_+0100__Received:_from_d06lmsgate_uk_ibm_com_(d06l
msgate_emea_ibm_com_[9_166_34_251])___by_d06relay03_por
tsmouth_uk_ibm_com_(8_12_9/NCO/VER6_5)_with_ESMTP_i
d_h53Lc6kZ273520___for_<fehners@uk_ibm_com>;_Tue,_3_J
un_2003_22:38:06_+0100__Received:_from_fundp_fundetective
_com_(fundp_fundetective_com_[66_111_221_66])___by_d06lm
sgate_uk_ibm_com_(8_12_9/8_12_8)_with_SMTP_id_h53Lb6D
R084210___for_<fehners@uk_ibm_com>;_Tue,_3_Jun_2003_2
2:37:56_+0100__Date:_06/03/2003_11:39:39_PM_GDT__Messa
geID:_<98095$ZmVobmVyc0B1ay5pYm0uY29t@fundp_fundet
ective_com>__From:_Norton_<info@fundetective_com>__Subj
ect:_Internet_Security_Privacy__Best_Price__ReplyTo:_info@f
undetective_comTo:_fehners@uk_ibm_com__Xtrack:_Itemize_
by_SMTP_Server_on_D06ML024/06/M/IBM(Release_5_0_9a_|
January_7,_2002)_at_03/06/2003_22:38:06__X-Mailer:_3_1_74-
XP/NG_[May_29_2003,_11:18:33]__Received:_from_d06relay0
3_portsmouth_uk_ibm_com_([9_180_34_41])_____________by_
d06ml024_portsmouth_uk_ibm_com
>msg856ec716_01f35_body0 1
__------------10546716279438__Content-Transfer-
Encoding:_7bit__Content-Type:_text/plain;_charset=us-
ascii;_classd=44:98j95joKYnh7wuM7_bM:2330019____Essenti
al_protection_from_viruses,_hackers,_and_privacy_threats____
___Symantec's_Norton_Internet_Security(tm)_2003_provides_e
ssential__protection___from_viruses,_hackers,_and_privacy_th
reats________Included_are_full_versions_of_Norton_AntiVirus
(tm)_and_Norton(tm)__Personal_Firewall,_which_efficiently_d
efend_your_PC_from_the_most__common_Internet_dangers__
You_also_get_Norton(tm)_Spam_Alert_to_block__unwanted_e
mail,_and_Norton(tm)_Parental_Control_to_protect_your__chi
ldren_online_______Get_it_today___http://fundetective_com:80
80/track?m=2330019&l=0&_e=4TSoBSPCDUgKewzKu6z.
…
…
…
.
.
```

```
>QUERY_MESSAGE
SRC="http://www_fundetective_com/images/security2003/sec2003_09_gif"_WIDTH=79_HEIGHT=60></TD><TD_COLSPAN=4>HREF="http://fundetective_com:8080/t
```

Fig. 1. A graphical representation of our SPAM-classification method. See also text for a detailed discussion.

When presented with an email message to classify (=query), we first generate the 'intersection' of the SPAM-vocabulary with the query sequence to find those patterns with instances somewhere in the email at hand: for each of the patterns that comprise the intersection we know both the pattern's instance in the query as well as the locations of all substrings in the knowledge-base that gave rise to the pattern in the first place.

At this point, there is substantial flexibility as to how this information is used to classify the query: even though a gamut of methods is possible, including Bayesian ones, we decided for the purposes of our prototype to keep things simple. In particular, we begin by setting up a vector of counters; the vector's length is equal to that of the query and the counters are all initialized to 0.

For each pattern *p* from the SPAM-vocabulary that has an instance in the query spanning positions q_{from} through q_{to}, we *add* a contribution CONTRIB(.,.) to the counters corresponding to region [q_{from}, q_{to}]. The manner in which we decide the value of CONTRIB(.,.) is described next.

2.3 The Method: contributing, scoring and thresholding

As we mentioned above, once the patterns from the SPAM-vocabulary with instances in the query-to-be-classified have been identified one can proceed in several ways. For example, patterns can be filtered using estimates of the probability that they can be random occurrences: a *k*-th order Markov chain can be built easily from the messages in the knowledge-base and subsequently used to generate probabilities for the patterns. A Bayesian scheme can be imposed. Etc. For the implementation of the Chung-Kwei prototype, we used the following simple scheme.

Recall that each pattern *p* that spans ℓ positions and is contained in the SPAM-vocabulary is a regular expression that has several instances in one or more email messages in the knowledge-base. Now if pattern *p* also has an instance in the query to be classified, then *p* will in fact pair up the fragment $q_i q_{i+1} \ldots q_{i+\ell-2} q_{i+\ell-1}$ (with $q_i = q_{from}$ and $q_{i+\ell-1} = q_{to}$) from the query and each fragment $m_j m_{j+1} \ldots m_{j+\ell-2} m_{j+\ell-1}$ that is an instance of *p* in the knowledge-base. Obviously, the more similar the query fragment is to each of the fragments in the knowledge-base the more likely it is that the query is in fact a SPAM message.

There is a lot of flexibility in the way one can assign scores to the vector of counters. In general, one can make use of a "scoring matrix" that generates contributions in a position-dependent and content-dependent manner. Let the set of printable characters that are in use for the language of interest comprise a total of *T* characters. We can then form a *T* x *T* matrix whose (*i*,*j*) location indicates the amount of similarity between the *i*-th and *j*-th characters; similarities could assume a value from a range of possible values. For example, and assuming that the character set in effect is the extended ASCII, cells of this matrix that could be given high similarity values might include (A,a), (B,b), (C,c), (D,d) etc. But it may also be beneficial to also give high values to the cells that correspond to pairs such as (A, Á), (i, í), (S, §), (u, ù), (e, ë), (c, ç), etc. which would of course allow us to determine that the two strings "**Atkins or the South beach diet**" and "**Átkíns or the §oùth bëaçh diet**" have a very high degree of similarity. Clearly variations of the above idea are possible.

Given such a scoring matrix, each instance of pattern *p* will contribute an amount CONTRIB(.,.) to the vector of counters that is determined as follows:

```
for $k$=1 to $\ell$ { counter_vector [$i+k-1$] += scoring_matrix[$q_{i+k-1}$][$m_{j+k-1}$] }
```

In other words, and for all values of *k*, the pattern *p* will contribute to the (i+k-1)-th position of the counter vector an amount that relates to the degree of similarity between the characters occupying the positions q_{i+k-1} and m_{j+k-1} respectively. The symbol += is a shorthand notation for 'increment by amount shown on the right of the = sign.'

Typically, the SPAM messages contained in the knowledge-base will be accumulated in an automatic manner. As such, it is entirely likely that the knowledge-base contains several messages with (nearly) identical headers and/or bodies. Consequently, each pattern *p* from the SPAM-vocabulary that has an instance in the query to be classified and involves these over-represented messages will lead to an artificial over-counting. However, bookkeeping can be introduced in a straightforward manner to prevent this from happening.

For the purpose of deciding whether a query message is SPAM, we make use of two criteria simultaneously: the number of patterns from the SPAM-vocabulary with instances in the query and the percentage of counters in the counter vector that have non-zero counts. The latter number is effectively equal to the portion of the query message that can be covered using patterns from the SPAM-vocabulary. Clearly, high numbers of patterns and high degree of coverage of a query message are desirable. This brings us to the issue of thresholding.

In the Chung-Kwei prototype two thresholds are used. One is T_p, the minimum required number of patterns with instances in the query, and the other is T_c, the minimum required coverage of the query. If the patterns from the SPAM-vocabulary that hit a given query message are such that both thresholds are exceeded, then the query message is reported as SPAM. These two thresholds are fixed ahead of time and are query independent. If a lot of SPAM messages are available and access to a repository of WHITE email is also possible, their values can be decided in an automated manner with an eye toward high sensitivity in identifying incoming SPAM messages and (near-)zero rates of false positives. We will pursue this further in the Results section.

3. Experimental Results

We have created a prototype implementation of Chung-Kwei and installed it on a 2.2 GHz Intel-Pentium PC in order to run the experiments that we describe here.

Chung-Kwei was first trained using a knowledge-base of accumulated SPAM email comprising 65,175 messages. As described above, our current implementation uses only the bodies of the SPAM messages to generate the patterns of the SPAM-vocabulary. We had originally set-up a knowledge-base containing 21,355 training-WHITE-mail messages to be used for ‘negative-training.’ However, we randomly-selected only one half of those WHITE messages and used it for negative training. The remaining half was used to decide the thresholds to be used by Chung-Kwei in the performance experiments. Those patterns from the SPAM-vocabulary that were also present in the one-half of WHITE-mail messages used for training were removed resulting in a final, cleaned-up collection comprising 6,660,116 patterns. With the SPAM-vocabulary at hand, we processed the bodies of the remaining one-half of the training-WHITE-mail messages, recording the number of patterns hitting and the percent of coverage for each message. The values for T_p and T_c were selected in such a way that the sensitivity of identifying a SPAM message would be maximum whereas the rate of false positives would not exceed 1 in 10,000; clearly, this step can easily be automated. For the knowledge-base that we used, the resulting values were T_p= 26 and T_c= 19%.

The entire training phase (pattern discovery + negative training + formation of final vocabulary + processing of WHITE mail to decide thresholds) was completed in ~17 minutes -- 1,050 seconds to be exact. During training, Chung-Kwei's memory requirements remained below 300 Mbytes.

Once the thresholds to be used were decided and fixed, we proceeded to test the system on a collection of 88,165 test messages: of these test cases, 21,198 messages were known to be WHITE email and 66,967 were known to be SPAM email. Our purpose here was two-fold. First, we wanted to determine the speed at which classification would take place. And second, we wanted to calculate the system's sensitivity and rate of false positives.

With respect to the performance timings, recall that the classifications in this implementation of Chung-Kwei are carried out by processing only the bodies of the SPAM and WHITE email messages. The average size of a SPAM message's body was 4.2 Kbytes whereas that of a WHITE message's body was 7.6 Kbytes. The total amount of processed bodies, SPAM and WHITE, was 422 Mbytes. Chung-Kwei processed the entire collection of 88,165 bodies in ~412 seconds, at an average of 214 messages/second. The implementation's memory requirements remained under 300 Mbytes of main memory throughout the classification phase.

Of the 66,967 SPAM email messages that were used for testing, Chung-Kwei correctly reports 64,665 as SPAM-email for a resulting sensitivity of 96.56%. Of the 21,198 WHITE email messages that were used for testing, Chung-Kwei reported 163 as being SPAM. Upon manual inspection of these 163 messages, we discovered that 137 were indeed SPAM messages that had erroneously been included in the WHITE mail collection! Of the remaining 26 messages, 12 were MIME-encoded and our decoder failed to translate them into plain-text; as a result, the message length was made artificially short resulting in an artificially high coverage that exceeded threshold: once these 12 MIME messages were manually decoded, Chung-Kwei correctly identified them as WHITE email. The remaining 14 messages were found to be bona fide WHITE email that was misclassified as SPAM; this resulted in a false-positive rate of 0.066% or one-in-six-thousand. Even though the false-positive rate is very low, it could have been made even lower had our negative training been based on the full 21,355 training-WHITE-mail messages instead of only one-half of them

4. Conclusions

We have presented Chung-Kwei, a new automated method for the automated classification of SPAM email. Chung-Kwei is based on a pattern-discovery framework that is combinatorial in nature. We have also run and discussed a number of experiments using Chung-Kwei and showed that the obtained sensitivity and rate of false positives are encouraging for this first iteration of the method.

Finally, the authors would like to thank Bob Filepp, Jason Crawford, Rich Segal, Jeff Kephart, V.T. Rajan and Mark Wegman for stimulating conversations during the last 12 months.

References

[1] Sahami, M., S. Dumais, D. Heckerman and E. Horvitz (1998) A Bayesian Approach to Filtering Junk E-Mail. Proceedings of AAAI-98 Workshop on Learning for Text Categorization. Madison, WI.
[2] Schleimer, S., D. Wilkerson and A. Aiken (2003) Winnowing: local algorithms for document fingerprinting. Proceedings of SIGMOD 2003. San Diego, CA.
[3] Yerazunis, W. (2004) The Spam-Filtering Accuracy Plateau at 99.9% Accuracy and How to Get Past It. MIT Spam Conference. Cambridge, MA.
[4] Damashek, M. (1995) Gauging similarity with N-grams: Language-independent categorization of text. Science, 267(5199):843—848.
[5] Rigoutsos, I., T. Huynh, A. Floratos, L. Parida and D. Platt, "Dictionary-driven Protein Annotation." Nucleic Acids Research, 30(17):3901-3916, 2002.
[6] Shibuya, T. and I. Rigoutsos (2002) Dictionary-driven Prokaryotic Gene Finding. Nucl. Acids Res., 30.
[7] Wespi, A., H. Debar, and M. Dacier (1999) An Intrusion-Detection System Based on the Teiresias Pattern-Discovery Algorithm. Proceedings EICAR'99. Aalborg, Denmark.
[8] Lillington, K. (1998) Teiresias on the Hacker Trail. Wired News. October 29.
[9] Rigoutsos, I. and A. Floratos (1998) Combinatorial pattern discovery in biological sequences: the TEIRESIAS algorithm. Bioinformatics, 14, 55-67.
[10] Rigoutsos, I. and A. Floratos (1998) Motif Discovery Without Alignment Or Enumeration. Proceedings 2nd Annual ACM International Conference on Computational Molecular Biology (RECOMB), New York, NY.
[11] Rigoutsos, I., A. Floratos, C. Ouzounis, Y. Gao and L. Parida (1999) Dictionary Building Via Unsupervised Hierarchical Motif Discovery In the Sequence Space Of Natural Proteins. Proteins: Struct. Funct. Genet., 37, 264-277.
[12] Rigoutsos, I., A. Floratos, L. Parida, Y. Gao and D. Platt (2000) The Emergence of Pattern Discovery Techniques in Computational Biology. Metabolic Engineering, 2,159-177.
[13] Floratos, A., I. Rigoutsos, L. Parida, G. Stolovitzky and Y. Gao (1999) Sequence Homology Detection Through Large-Scale Pattern Discovery. In Proceedings Third Annual ACM International Conference on Computational Molecular Biology (RECOMB '99), Lyon, France.
[14] Floratos, A., I. Rigoutsos, L. Parida and Y. Gao (2001) DELPHI: A pattern-based method for detecting sequence similarity. IBM Jrnl of Research and Development. 45, 455-474.
[15] Rigoutsos, I., P. Riek, R. M. Graham and J. Novotny (2003) Structural Details (Kinks and Non-a Conformations) in Transmembrane Helices are Intrahelically Determined and can be Predicted by Sequence Pattern Descriptors. Nucleic Acids Research, 31(15):4625-31.
[16] Stolfo, S. J., S. Hershkop, K. Wang, O. Nimerkern and C. Hu (2003) A Behavior-based Approach to Securing Email Systems. Mathematical Methods, Models and Architectures for Computer Networks Security, Springer Verlag.