

The Computer and Natural Language (Ling 445/515)

Topic 3: SPAM detection

Markus Dickinson

Dept. of Linguistics, Indiana
Autumn 2008

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

Introduction

- The issue

- Social context

- Rule-based approaches

- Statistical approaches

Spam Issues

- Tokenization

- Devious spam

Practical aspects

Introduction

- The issue

- Social context

- Rule-based approaches

- Statistical approaches

Spam Issues

- Tokenization

- Devious spam

Practical aspects

- ▶ Text classification techniques generally carries over to identifying spam.
- ▶ **spam** = e-mail we don't want, usually only loosely directed to us, including unsolicited commercial e-mail
- ▶ Structure of discussion:
 - ▶ The issue and its social context
 - ▶ Language technology: rule and statistical methods
 - ▶ Further issues: tokenization & devious spam
 - ▶ What you can do about spam

A large part of the discussion is based on Jonathan A. Zdziarski's excellent 2005 book *Ending Spam*

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

- ▶ Spam consumes
 - ▶ a significant fraction of total Internet bandwidth, which causes both a slowdown of other traffic, and possibly raises overall bandwidth cost.
 - ▶ a large amount of storage space on mail servers, sometimes actually making it temporarily impossible for legitimate messages to be received.
 - ▶ a significant portion of the time and effort of people who use email to communicate.
- ▶ Spam can be the vehicle of identity theft campaigns, other types of fraud, and virus propagation.
 - ▶ See <http://www.sec.gov/answers/nigeria.htm> for an explanation of Nigerian Advance Fee Fraud.

(based on *Spam: The Phenomenon* by Colin Fahey,
http://www.colinfahey.com/oldpages/spam_topics/index.htm)

- ▶ A spammer obtains email addresses, e.g., by sending out robots to collect e-mail addresses from web-sites and newsgroups, or by buying (legally or illegally created) address databases
- ▶ To that collection of addresses, the spammer often automatically generates other possibilities.
 - ▶ e.g., “I’ve found `abc1@indiana.edu` and `abc23@indiana.edu`. What if I try other `abc#@indiana.edu` combinations?”
- ▶ A message is sent out. The spammers are aware of various filters and so try to make their messages devious.

(cf. <http://www.philb.com/spamex.htm>)

A brief timeline of the history of spam (cf. Zdziarski, ch. 1):

- ▶ 1978: First spam sent over the Arpanet (internet precursor), advertising for Digital Equipment Corporation
- ▶ 1982: First email chain letter
- ▶ 1988: *Jay-Jay's College Fund*: a student in Nebraska sent a message to different newsgroups asking for money
- ▶ 1994 (Jan.): First widespread spam, the “Jesus” spam, posted to every newsgroup on Usenet

Some spam history (cont.)

- ▶ 1994 (Apr.): First bulk mailer software, courtesy of the Canter & Siegel spam: husband & wife attorneys hired a programmer to post ad to every newsgroup
- ▶ 1994 (Dec.): First “spam-fighting superhero” (Zdziarski, p. 14), the anonymous Cancelmoose, who cancelled Usenet postings s/he believed were spam (based on how many newsgroups messages posted to)
- ▶ 1995: Jeff Slaton (the “Spam King”), “Krazy” Kevin Lipsitz, and Stanford (“Spamford”) Wallace become notorious spammers
 - ▶ Issues of free speech & ignorance of companies using spam are prominent
- ▶ 1995: First commercial spamware for people to send spam, Floodgate

Some spam history (cont.)

- ▶ 1995 (Aug.): First known list of public email addresses (2 million addresses)
- ▶ 1995 (Nov.): First “remove lists” created
- ▶ 1996: First blacklist, Spamhaus
- ▶ 1997-2000: spam grows & war on spam starts
- ▶ 2000: First Nigerian spam, dubbed a 419 spam, after the Criminal Code of Nigeria banning this fraud
- ▶ 2001+: spam grows exponentially

- ▶ Spammers are trying to make money by selling a product
- ▶ Sending email is virtually free, even if millions of messages are sent
- ▶ Enough people fall for spam to make it worthwhile

- ▶ But the negative consequences of spam on our resources are well-established, so how can the problem be addressed?
 - ▶ Laws don't seem to work well: spammers use other countries, are hard to trace.
 - ▶ Checking to see if a human is on the other end before accepting an e-mail takes extra time and effort.
 - ▶ Charging for e-mails would mean the end to e-mail as we know it.

Some Approaches to Fighting Spam

Some general ways to combat spam (cf. Zdziarski, ch. 2):

- ▶ Blacklisting, or blackholing: maintain a list of networks (ISPs) known to send (only) spam
 - ▶ *Problems*: Only works after spam is sent; hard to maintain quality
- ▶ Whitelisting: only accept mail from known people
 - ▶ *Problems*: Rejects legitimate mail; addresses can be forged
- ▶ Challenge/Response: like whitelisting, but allow new addresses to prove that they come from humans
 - ▶ *Problems*: Results in more email traffic; some people dislike it; can be forged
- ▶ Throttling: slow down the rate of inbound/outbound emails on a network, severely slowing down bulk mail
 - ▶ If a message looks like spam, the throttling system will put the spammer on hold indefinitely
 - ▶ *Problem*: Relies on spam filtering

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

Some Approaches to Fighting Spam

Litigation

Can be hard to track spammers, but ...

- ▶ Unspam (<http://www.unspam.com>): consulting company to help in litigation
 - ▶ Child Protection laws are one popular way to do this
- ▶ Project Honey Pot (<http://www.projecthoneypot.org/>)
 - ▶ “Project Honey Pot is the first and only distributed system for identifying spammers and the spambots they use to scrape addresses from your website. Using the Project Honey Pot system you can install addresses that are custom-tagged to the time and IP address of a visitor to your site. If one of these addresses begins receiving email we not only can tell that the messages are spam, but also the exact moment when the address was harvested and the IP address that gathered it.” (http://www.projecthoneypot.org/about_us.php, accessed 10/2/08)
- ▶ Spammer fingerprinting also done to identify the authors

Some Approaches to Fighting Spam

Primitive Language Analysis & Heuristic Filtering

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

All of the previous approaches rely mainly on origin of spam, but we will focus on the content of spam

- ▶ Primitive Language Analysis: identify key words which are indicative of spam (e.g., *Click here*)
 - ▶ *Problems*: not very accurate; flag lots of non-spam
 - ▶ Still, provides a starting point for content analysis
- ▶ Heuristic filtering: like above, but also have rules for non-spam and can weight items
 - ▶ This is rule-based filtering, which we turn to now ...

Rule-based filtering = filtering e-mail based on set rules.

Rule-based spam filters can be rather sophisticated:

- ▶ can **weight** patterns detected by the rules:
 - ▶ e.g., 3 points for *viagra* in the header, 2 for originating from a hotmail account, -2 points for a “.edu” address, ...
 - ⇒ When you pass some threshold of points, it's marked as spam.
- ▶ can use information about systems it knows about:
 - ▶ e.g., This html message came from Outlook, but Outlook can't send pure html messages

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

Spam example

Spam detection software (here: spamassassin) has identified this incoming email as possible spam. It provides:

- ▶ Content preview:

Email Marketing Email more than 2,500,000+ TARGETED prospects EVERYDAY! That's over 75,000,000+ prospects per month (and growing!). Our Optin email safelists are 100% Optin and 100% legal to use. Your ad will reach only those prospects who have requested to be included in Optin safelists for people interested in new business opportunities, products and services. [...]

- ▶ Content analysis details: (11.2 points, 5.0 required)

Some Rules

pts	rule name	description
0.1	HTML-TAG-EXISTS-TBODY	BODY: HTML has “tbody” tag
0.1	HTML-FONTCOLOR-RED	BODY: HTML font color is red
0.1	HTML-FONTCOLOR-BLUE	BODY: HTML font color is blue
1.6	FORGED-MUA-OUTLOOK	Forged mail pretending to be from MS Outlook
1.1	FORGED-OUTLOOK-TAGS	Outlook can't send HTML in this format
0.0	CLICK-BELOW	Asks you to click below
1.9	MIME-HEADER-CTYPE-ONLY	'Content-Type' found without required MIME headers
1.7	HTML-MIME-NO-HTML-TAG	HTML-only message, but there is no HTML tag
1.1	FORGED-OUTLOOK-HTML	Outlook can't send HTML message only

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

Rule-based filters are quite intuitive and can be highly effective, but they also have drawbacks:

- ▶ Someone has to identify a pattern and specify a rule matching it (with high precision/recall).
- ▶ The more rules there are, the better it detects, but the slower it runs.
- ▶ The rules are designed for everyone's inboxes, but all inboxes are different
- ▶ Rule-based filters by nature are a step behind the spammers:
 - ▶ rules can only be developed once a pattern has been observed in spam, and
 - ▶ once a spammer knows a rule, they will can try to bypass it.

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

- ▶ Statistical filters have been proposed in place of or in addition to rule based ones.
- ▶ Instead of providing hand-written rules, one provides large sets of examples, one set with messages known to be spam, another with messages known to be ham.
- ▶ How it works:
 - ▶ Count up occurrences of words in previous e-mails:
 - ▶ How many times does X appear in something flagged as spam?
 - ▶ How many times does X appear in something which isn't spam? (i.e., is ham)
 - ▶ From these counts, we calculate the **spam probability** of a word.

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

Calculating probability (1)

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

- ▶ Setup
 - ▶ For training our spam filter, we saved 1500 messages, 1000 spam mails and 500 real e-mails. But, since there were many emails we didn't save, we won't assume that this spam/ham proportion is accurate.
 - ▶ *cash* appears in 203 e-mails, 200 of which are spam, 3 of which are real.
- ▶ So, in 20% of spam messages (200/1000), *cash* appears, while it appears in only 0.6% of real messages (3/500). That is,
 - ▶ $Pr(\text{cash}|\text{spam}) = 0.20$
 - ▶ $Pr(\text{cash}|\text{ham}) = 0.006$

Calculating probability (2)

What we want to find is: $Pr(spam|cash)$

We'll talk about Bayes' Law later in the semester, but it boils down to the following:

$$Pr(spam|cash) = \frac{Pr(cash|spam)Pr(spam)}{Pr(cash)}$$

The $Pr(cash)$ can be expanded because it either comes from a *spam* message or a *ham* one (no other options)

$$Pr(spam|cash) = \frac{Pr(cash|spam)Pr(spam)}{Pr(cash|spam)Pr(spam)+Pr(cash|ham)Pr(ham)}$$

Calculating probability (3)

Now suppose we don't know $Pr(spam)$. If we just assume $Pr(spam) = Pr(ham) = 0.5$, we get:

$$Pr(spam|cash) = \frac{Pr(cash|spam)}{Pr(cash|spam) + Pr(cash|ham)}$$

And recall that we already calculated:

- ▶ $Pr(cash|spam) = 0.20$
- ▶ $Pr(cash|ham) = 0.006$

So, we wind up with:

$$\begin{aligned} Pr(spam|cash) &= \frac{Pr(cash|spam)}{Pr(cash|spam)+Pr(cash|ham)} \\ &= \frac{0.20}{0.20+0.006} \\ &= 0.971 \end{aligned}$$

- ▶ We calculate this probability for every word.
- ▶ When a new e-mail comes in, we extract all the words and find their probabilities.
- ▶ We pick the 15 (or so) words which are the best and the worst indicators of spam (farthest from the middle)
 - ▶ i.e., Pick the 15 words which give the strongest indication as to the true contents of the message.
- ▶ Combine these probabilities into a single probability
- ▶ If the probability is high enough (maybe 90% or more), call it spam.

Spam detection example

So, let's say that you get an e-mail from me saying:

Hey, class, I just heard about a great opportunity in Nigeria to study and even make money.

...

I've also put a quiz on-line and asked one of the linguistics students to take it for a test drive so we can be pretty sure it works.

Markus

Spam detection example (2)

- ▶ We extract words with high probabilities of being spam: *opportunity*, *Nigeria*, *money*, ...
- ▶ and words with low probabilities of being spam: *linguist*, *quiz*, and possibly *Markus* [it's often hard to realistically fake an acquaintance's name]

We combine these probabilities, and it turns out that *opportunity* and *money* are indicators of spam, but *quiz* and *linguistics* are very good indicators of non-spam.

Just for reference, the formula for combining the 15 most extreme probabilities is:

$$(1) \Pr(\text{spam}) = \frac{P_1 \times \dots \times P_{15}}{P_1 \times \dots \times P_{15} + (1 - P_1) \times \dots \times (1 - P_{15})}$$

Note that at some point, this non-spam e-mail will itself be used in recalculating probabilities for words.

- ▶ That is, the spam filter is continually **learning** what is spam and thus adapting to new spam techniques
- ▶ As with general document classification, this idea of **machine learning** is very important & widely-used.

Machine learning = computer learns how to behave based on previously-seen data.

Some perks of statistical filtering

Paul Graham (<http://www.paulgraham.com/wfks.html>) list of the benefits of statistical filters:

1. They're effective: they tend to catch 99% of spam.
2. They generate few **false positives** = real e-mails mistakenly treated as spam
3. They learn.
4. They let the user define what spam is → one person's spam is another person's golden opportunity
5. They're hard to trick: to fake the statistical filters: use fewer bad words, or use more innocent words.
 - ▶ But the innocent words are defined by the user

Improving spam filtering

It turns out that spam filters can be 5-10 times more accurate than humans (Zdziarski, p. 48)

Still, improvements to be made (cf. Zdziarski, ch. 3-4):

- ▶ Gary Robinson: took Graham's model and made low-frequency tokens have less of an impact
 - ▶ Intuition: just because a word appears only in one spam message doesn't mean it's a 100% spam indicator
 - ▶ Technique is better at preventing false positives
 - ▶ Also developed techniques to allow for *uncertain* classification
- ▶ Brian Burton (SpamProbe): look at 27 elements (instead of 15)
 - ▶ Allow words to count multiple times if in an email multiple times
- ▶ Iterative training: if a filter misclassifies a message, it'll retrain on it
 - ▶ The filter retrains until it gets it right

One newer approach to spam filtering is **collaborative filtering** (Zdziarski, ch. 14)

- ▶ Statistical filters are great, in that they learn my personal preferences
- ▶ But can't we learn something about spam collectively?

Message inoculation (Bill Yerazunis) works by “vaccinating” people from spam messages

- ▶ If one user gets a new spam message which passes by their filter, they pass this message on to their friends, but, crucially, marked as spam
- ▶ The other users are able to train their filter on this spam without worrying about a similar spam showing up as a false positive

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

One issue we've largely ignored is that of tokenization:
breaking messages down into component words

This can dramatically affect accuracy

- ▶ Intuitively, text is tokenized into words
- ▶ But it's often crucial to know where the text came from, so we get **structured information** like:
 - ▶ Subject*Free!
 - ▶ Url*indiana
 - ▶ Url*edu
 - ▶ From*abc@indiana.edu [note how this can create blacklists/whitelists]
 - ▶ Hello [unstructured = simple text in body of email]
- ▶ We'll also have to deal with HTML markup, e.g., HTML tags like *applet* are potentially spam

cf. Zdziarski, ch. 6

- ▶ Tokenize text into words, but whitespace isn't enough
 - ▶ Should *free* be treated differently than *free!* or *free!!*?
 - ▶ Common solution: treat one exclamation as part of the word, but the rest are ignored (i.e., *free!* & *free!!* treated the same)
- ▶ Have to reassemble some words
 - ▶ e.g., *I/T/S F_R_E_E* is probably better seen as *ITS FREE*
 - ▶ However, sometimes (common) partial words such as *VIA* or *GRA* can be spammy
- ▶ Can take a token and “degenerate” it until it matches something in memory
 - ▶ e.g., never seen *Subject*Free?*, but can remove distinction until we see that we have seen *free*
- ▶ Can also use *n*-grams of words: more accurate, but more storage

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

- ▶ Spam filters try to distinguish spam from ham, using rules and patterns of word occurrences that it has learned about.
- ▶ Spammers want to disguise their messages so that they trigger none (or only few) of the rules and do not contain occurrences of words typical for spam.
- ▶ Emails are often encoded in HTML (hypertext markup language), so we need to talk about this encoding before we can take a closer look at various spammer tricks.

Zdziarski's book (ch. 7) and a website by John Graham-Cumming, <http://www.jgc.org/tsc.html> (*The Spammers' Compendium*), are good sources for devious spam.

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

The Hypertext Markup Language (HTML) provides meta-information which tells a web browser or mail reader how a document is structured and how it should be displayed.

- ▶ HTML markup has beginning and end tags
 - ▶ `Example`: tells the browser to render the text Example in bold, i.e. as **Example**
- ▶ An HTML tag can have attributes
 - ▶ For example, *color* is an attribute of the *font* tag.
 - ▶ `Language` makes *Language* appear blue

Tokenization changes

Text splitting

Make words which are good indicators for spam look less like words:

- ▶ Space out words to make them unrecognizable to word detectors
e.g., M O R T G A G E
- ▶ Other characters can be used instead to space things out
e.g., F*R*E*E V'I'A'G'R'A O!NL#\$N%E

Solutions:

- ▶ Reassemble split words
- ▶ Trust the filter to learn important patterns from small pieces of data (who else uses so many capital Vs?)

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

Tokenization changes

Symbolic text

If you can alter characters, words won't appear as the same words which are frequently found in spam.

- ▶ Replace letters that look like numbers with numbers
e.g., V1DE0 T4PE M0RTG4GE
- ▶ Use accented characters in English
e.g., Fántàstìc – earn mōnégý thrôugh unçõlleçtéd judgments

Solutions:

- ▶ Undo these mappings
- ▶ Again, trust the filter: these are very clearly spam words

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

Tokenization changes

“Hypertextus interruptus”

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

Again, spammers try to make spam words hidden

- ▶ Make it so that a single suspect word isn't seen as a single word by the detector—but it is seen by the human as a single word.
 - ▶ e.g., milli<! xe64 >onaire

Solution:

- ▶ Reassemble such split up words, requiring filters to understand HTML very well

Tokenization changes

Table-based obfuscation

One especially devious tactic involves taking English text and dividing it vertically

- ▶ Take the English text and instead of printing it out horizontally, print it vertically in a table
- ▶ The result will look like English to the user, but will only be word fragments to the parser.

Solutions:

- ▶ Make the filter see what the human sees, i.e., piece the table together (but this is costly)
- ▶ Or, again trust the filter to find weird combinations of characters to be indicative of spam

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

Tokenization changes

ASCII spam

A fairly recent spamming technique is to draw pictures using regular ASCII characters

- ▶ To a human, it appears as a picture
- ▶ To a computer, it's meaningless sequences of characters

Luckily, the spammer often also includes something like a URL address, indicating spam

Solutions:

- ▶ It's possible that filters will catch these anyway, due to URLs and header tokens.
- ▶ Hard to tell at this point what should be done.

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

Statistical attacks

Bayesian poisoning

Bayesian poisoning tries to mess up the probabilities for a statistical filter:

- ▶ Spammer sends you a number of fairly innocuous (or even empty) emails, e.g., *Thanks for your help!*
- ▶ Hidden in that email is a header with garbage for content, e.g., *X-Wadferg0: klijkw wereag jipwe nzcxgow asdf*
- ▶ Assuming you do not tell your filter that these are spam, then your filter learns these “words” as legitimate data
- ▶ Then spammer sends you a real spam, including these legitimate words

Solutions:

- ▶ Always mark mysterious messages as spam (and check headers, if necessary)
- ▶ More sophisticated header analysis

Flooding the filter

Invisible Ink

Spammers do things which can mess up your spam filter by secretly including words which make the e-mail sound legitimate, but which the e-mail user never sees.

- ▶ Add some real random words before HTML.
suspensory obscure aristocratical meningorachidian
unafeared brahmachari
<html>
- ▶ Write white text on a white background
suspensory obscure aristocratical
meningorachidian unafeared brahmachari

Solutions:

- ▶ Include in calculation exactly what the users sees
- ▶ Trust that the filter will treat these unknown words as neutral words until it knows better

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

Flooding the filter

Hiding the contents in other media

- ▶ Instead of encoding a message in a text, spammers
 - ▶ send images
 - ▶ send http links to images

Note: By having each spam message load a different image name, the image loading can function as a message to the spammer signaling this message has been read.
 - ▶ send programs (javascript), which when executed get the text from another computer, essentially loading a web page
- ▶ Relies on the mail reader to be able to display images and execute programs.

Solutions:

- ▶ Very hard to detect as spam, but since the use of these features for benign purposes is not common, one can just switch off the loading of images and deny execution of programs in general.

What you can do about spam

Negatives

- ▶ Don't ever buy anything advertised through spam—if everyone observed this, spamming would not pay off and stop existing.
- ▶ Be careful about:
 - ▶ Asking to be taken off a list.
Clicking on “remove me,” or replying to spam mail will let them know your e-mail is valid.
 - ▶ Posting to a newsgroup which publicly archives their messages
 - ▶ Marking (or, more likely, not unmarking) that box when signing up for an account which says something like “I'd like to receive offers . . . ”
 - ▶ Posting your e-mail on your website or in newsgroups.

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects

What you can do about spam

Positives

- ▶ Things you can do:
 - ▶ Create accounts specifically used for newsgroups and such
 - ▶ Make your e-mail address on your website readable *only* to humans.
e.g., abc2ATindianaPERIOD—and don't forget that “edu” at the end
 - ▶ Use a properly configured spam filter (e.g., the free spamassassin is very well configurable)
 - ▶ Avoid loading images in emails by default, if possible.

Introduction

The issue

Social context

Rule-based approaches

Statistical approaches

Spam Issues

Tokenization

Devious spam

Practical aspects