KNOWLEDGE MODELING OF PHISHING EMAILS

A Thesis Proposal

Submitted to the Faculty

of

Purdue University

by

Courtney Falk

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

August 2016

Purdue University

West Lafayette, Indiana

Dedicated to Michelle, Sophia, and Alexander

ACKNOWLEDGMENTS

Thanks to my committee; doctors Victor Raskin, Julia Taylor, Eric Dietz, and John Springer for their assistance and guidance of this research.

Thanks also goes to doctors Melissa Dark, Dongyan Xu, and Xiangyu Zhang for the financial support that their assitantships brought.

And special thanks to my family, without whom this wouldn't be possible.

PREFACE

This dissertation grew out of an enthusiasm for linguistics in general and computational linguistics specifically that I discovered while working on my masters degree.

The core of this work centers around Ontological Semantics Technology, a form of natural language processing which uses an ontology to structure its semantic meaning. I became interested in how this might leverage other, parallel developments in machine learning. To that end I began a series of experiments to determine empirically how ontology-based natural language processing performs when compared to established approaches used in statistical machine learning.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

ABBREVIATIONS

AI	Artificial intelligence
AOL	America Online
APWG	Anti-Phishing Working Group
CVE	Common Vulnerabilities and Exposures
DNS	Domain Name System
FOIA	Freedom of Information Act
FP	False Positive, or type one error
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IP	Internet Protocol
KB	Knowledge base
LDC	Linguistic Data Consortium
LSA	Latent Semantic Analysis
ML	Machine Learning
MBML	Meaning-Based Machine Learning
MIME	Multipurpose Internet Mail Extensions
MP3	MPEG Audio Layer III
NIST	National Institute of Standards and Technology
NLP	Natural language processing
OST	Ontological Semantics Technology
PDF	Portable Document Format
PII	Personally identifiable information
POP3	Post Office Protocol version 3
POS	Part Of Speech

SMTP	Simple Mail Transfer Protocol
SVM	Support Vector Machine
TF-IDF	Term Frequency, Inverse Document Frequency
TP	True Positive
TMR	Text Meaning Representation
TREC	Text REtrieval Conference
URL	Uniform Resource Locater

NOMENCLATURE

email	electronic mail
OntoSem	Ontological Semantics
OPSEC	Operations Security

GLOSSARY

con	a confidence scam or fraud
ham	known good email, as in opposition to spam
PE32	32-bit executable Windows binary file format
scraping	copying the content and structure of an existing web page
spam	unsolicited sales offer usually via email
x86	32-bit Intel processor architecture

ABSTRACT

Falk, Courtney Ph.D., Purdue University, August 2016. Knowledge Modeling of Phishing Emails. Major Professors: Victor Raskin and Julia Taylor.

This dissertation describes the knowledge modeling necessary to semantically understand phishing emails. Samples of both good, non-malicious emails and phishing, malicious emails are processed manually. The results of the processing is analyzed for indicators in the semantic structure that might aid in the identification of phishing emails.

CHAPTER 1. INTRODUCTION

One would soon go mad if one took such coincidences too seriously. One might be led to suspect that there were all sorts of things going on in the Universe which he or she did not thoroughly understand.

Kurt Vonnegut

Bluebeard

This chapter introduces the reader to the basic concepts involved with the problem of phishing, how phishing attacks work, and the current scope of the problem.

Trends in cyber attacks ebb and flow. But phishing is a class of cyber attacks that spans these trends. What makes phishing a perennially attractive choice for attackers? The reason is because the key component to the success of any phishing attack, the human user, will always be reachable. Firewalls can stop remote exploits from reaching vulnerable servers, and armed guards are a good deterrent against physical intruders, but humans need to communicate in order to work effectively and these communications channels are all the opening needed to create a successful phishing attack.

1.1 Definition

In order to sign up for America Online (AOL) a person must provide a credit card number. But in the early days of AOL there was a flaw in their account creation system; the credit card numbers were not confirmed as being active. This meant that someone could create a valid credit card number, one that didn't match an existing account, and use it to create an AOL account. Even in the early 1990s the software existed and was widely available to generate credit card numbers. Hackers would use this technique to get initial access to online services and then use that first account to pivot, stealing other, legitimate accounts in order to maintain an online presence. It was these stolen accounts of paying members that were called "phish", and the act of stealing the accounts was called "phishing."

In January of 1996, user "Weapon X" on the alt.2600 newsgroup explained the motivation of the phishing attack sequence:

> First off, WHY would you want to go back to AOL once you get a real inet acct? I was stupid enough to bother with that fake acct/phishing period back in the day, but right now, from what i hear, the best way is to use the generated cc#s, and use the maybe 5-10 minutes you have, and go phishing... Then from that phish, keep phishing from that one (use side accts of coarse [sic]...) Weapon X (1996)

Eventually manual phishing on AOL was automated thanks to the AOHell tool (Langberg, 1995). AOHell offered hackers a graphical interface tool to facilitate multiple different acts of mischief on AOL, not the least of which was the phishing for other users' credentials.

The term "phishing" creates a vivid analogy: The attacker is the fisherman, casting a baited hook out to where he suspects it will be noticed by victims. These victims are the fish, entranced by a shiny bauble and suckered into taking the bait. When the fish bite onto the hook then the angler reels them in to complete the attack.

Even the way that "phishing" is spelled gives clues to its origin. The correct spelling is with an "f", pronounced as a fricative. There are places in English orthography where the "ph" consonant cluster also is pronounced as a fricative, but in these cases it suggests a Greek origin for the word. "Fishing" is not one of those words. So why the spelling tweak?

Phreaking was a hacker activity that predates phishing by decades going all the way back into the 1970s when “Captain Crunch” (John Draper), Steve Jobs, and Steve Wozniak were creating hardware to defraud the telephone companies of free long distance service (Wozniak & Smith, 2006).

The term, “phreaking,” is a portmanteau of “phone” and “freaking”. Over time hackers began substituting “ph” for “f” in arbitrary places, creating leetspeak slang. But in the case of phishing the “ph” has a stronger tie back to the original intent of phreaking as an inherently social activity that requires reaching out to other human users (Sterling, 1993).

Over time semantic drift caused the meaning of the phish- neologisms to change. “Phishing” now means the act of using social engineering to coerce a user into giving up personal information or access to personal resources.

Social engineering itself is just a modern day confidence scam with successful conmen like Kevin Mitnick gaining celebrity status within both the hacker and information security communities (Mitnick & Simon, 2002).

1.2 Unsolicited Email

Phishing emails are sometimes discussed in conjunction with spam emails (James, 2005). The two categories do share one unifying attribute in that they are both unsolicited contacts. But there are substantive differences between the two.

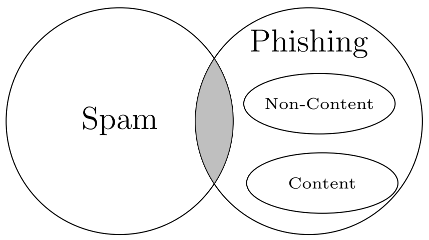


Figure 1.1. The relationship between spam and phishing concepts.

The key different between spam emails and phishing attack emails is the intended end action on the part of the targeted user. With spam emails a company hires an intermediary spam marketer to deluge large numbers of email addresses with unsolicited offers for products and services. Some of these products and services are of dubious legality and quality. But the spammer only cares about the raw number of spam emails set and their conversion rate into hits on the client's web site.

Phishers, as opposed to spammers, are serving their own goals. If a phisher sends out a phishing email then he/she wants to collect the targeted user's information or resources for personal use. Sometimes the attack itself belies the intended use of the user's information or resource. A phishing email that requests details of financial accounts likely wants to use that information to steal money out of the account in question. Whereas a phishing email that tries to exploit a software vulnerability on a user's computer wants the resources that come with being able to run software on someone else's machine; bandwidth and disk space to name two.

1.2.1 Spam

Spam is the unsolicited emailing of offers for products and services. The users receiving the spam emails may not be of the target audience for the products and services in question.

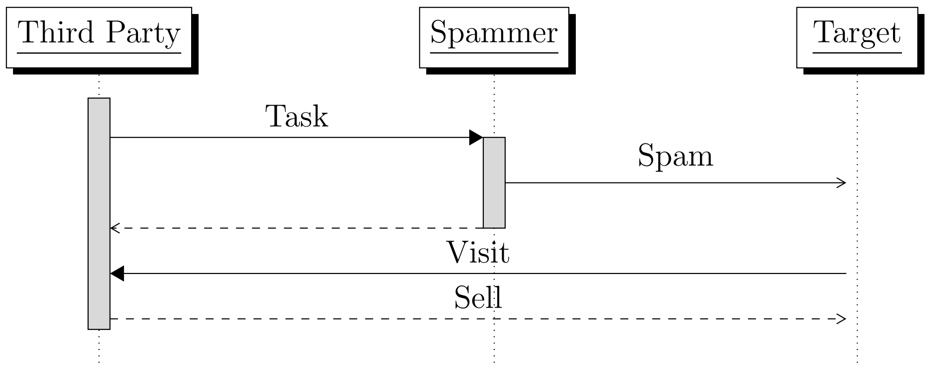


Figure 1.2. The sequence of events in a spam campaign.

Figure 1.2 shows the events and messages involved in an end-to-end spam email campaign. In this example, a third party has hired a spammer to send unsolicited product offers to email addresses across the Internet. The goal is to entice some number of spam recipients into paying customers of the third party. The target user doesn't need to ever interact again with the spammer again.

1.2.2 Phishing for Credentials

The theft of user login credentials is where phishing began. Once an attacker had a user's login name and password the attacker could log in as that user and steal whatever resources that account might have access to. In the case of the AOL account theft described previously, the attacker wants to steal Internet access.

With the advent of online banking came the specter of phishing for bank login credentials (Sangani, 2003). Once an attacker can log into a bank online as a victim there is very little keeping the attacker from transferring the entire balance of the victim's accounts to accounts that belong to the attacker. This fraud is sometimes reimbursed by the victim's bank. The reimbursement costs get passed on to all the users of the bank by way of increased fees.

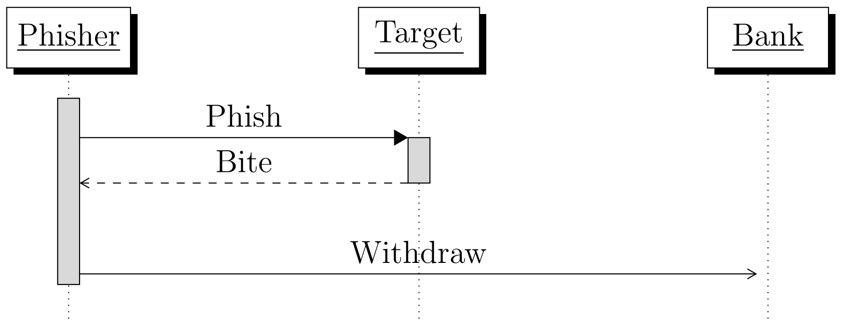


Figure 1.3. Phishing for account information.

Figure 1.3 makes it immediately obvious what the difference between spamming and phishing. The phisher wants to steal a target user's information for the purposes of making fraudulent withdrawals from the target's bank account.

1.2.3 Phishing to Exploit Computers

As hackers became more savvy about computer security and reverse engineering they began to better exploit software vulnerabilities. Then the underground hacker magazine, Phrack, released an article with detailed instructions on how buffer overflow vulnerabilities could be exploited (Aleph One, 1996). This led to a proliferation in exploits for vulnerabilities in software of all types.

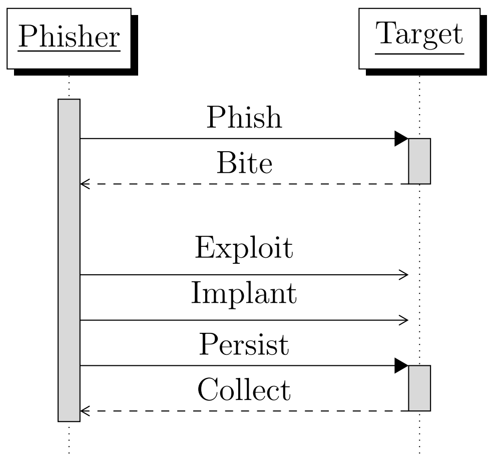


Figure 1.4. Phishing for computing resources.

There is a difference in the technical requirements for a phishing attack that only wants to steal information and a phishing attack that targets computing resources. The former interacts only with the human user and can use the same phishing script through email, instant message, or social network. A phishing attack that relies on malware to steal computing resources has added technical requirements. An exploit requires certain operating system and software versions, and can be defeated by anti-virus software and other defensive measures. Both types of phishers need to have at least an understanding of the targets' language and culture.

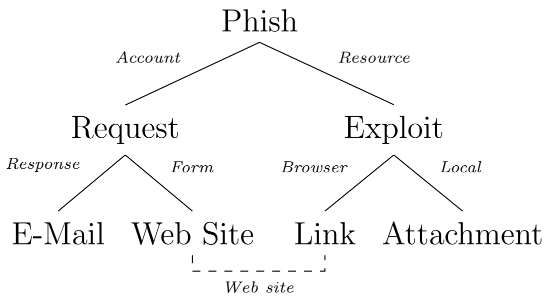


Figure 1.5. A brief taxonomy of four types of phishing attacks.

Figure 1.5 shows a taxonomy of how different types of phishing attacks are related. At the top level of the taxonomy, the distinction between Request and Exploit phishing attacks is that the Request attack is targeting account information ("Account") while the Exploit attack wants to appropriate someone else's computer resources ("Resource").

Request-type attacks divide into E-Mail and Web Site subtypes. The differentiating factor is how the attacker desires for the victim to complete the attack. In E-Mail attacks the victim writes a response email with the included account information. In Web Site attacks the victim follows a URL to a web page that is mocked up to look legitimate. For instance, if the phishing email claims to be from Bank of America then the URL will lead to a phony web page made to look like the actual Bank of America login page. The victim enters his/her account credentials and the attack is complete.

The Exploit attack is subdivided into Link and Attachment types. Link-type exploit attacks try to entice a user to follow a URL in the email that opens a malicious web page which will in turn attempt to exploit that user's web browser (Howard, 2012). Local-type exploit attacks include the exploit code as a component of the message either encoded in the body text or as an attachment, depending on the type of vulnerability targeted by the exploit.

Figure 1.5 includes a dotted line between the Web Site and Link attack subtypes. This signifies that both types of attacks lure the victim to a web site. The dotted line then indicates this commonality between the two attacks. The difference between Web Site and Link attacks is explained in the taxonomy by how they branch off from the different Request and Exploit attack types respectively. But these two attacks are not mutually exclusive. In fact, a savvy phisher might include both forms of attack on the same web page so that even if a victim gets cold feet before entering the account information there is still the chance that the embedded exploit has worked, giving the phisher remote access to the victim's computer.

1.3 Quantifying The Problem

Several public security companies record and report data concerning phishing attacks. The trends that these reports describe is not encouraging.

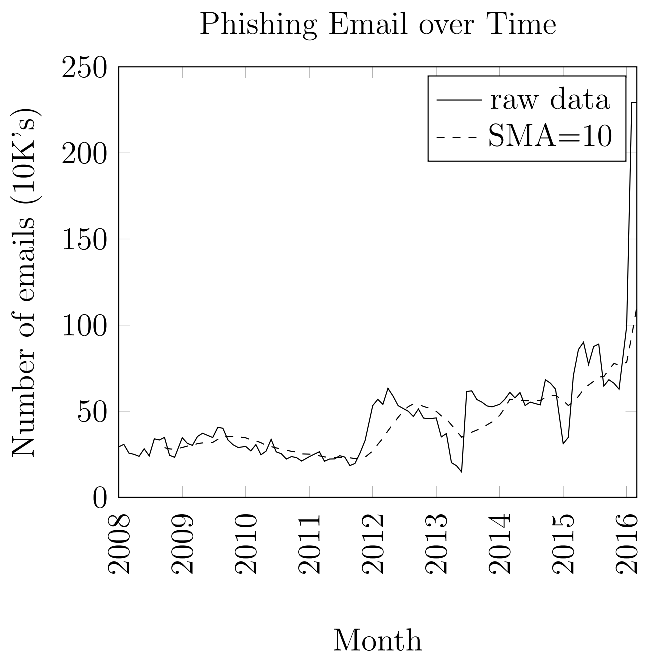


Figure 1.6. Number of reported phishing emails collected by APWG.

The chart in Figure 1.6 shows the trend of phishing emails reported to APWG in the period beginning first quarter of 2008 and ending after the first quarter of 2016. The numbers reported by (RSA, 2014) for the same period of 1Q 2013 show an increase in phishing activity, which contradicts the APWG data. Allowing for a certain amount of volatility still highlights a trend of increasing number of phishing attacks.

One tool utilized by phishers is the PDF exploit. The PDF document format is a popular vector for several reasons. First, the design goal of PDF is to be cross-platform so PDF readers exist for every major operating system, including mobile operating systems. Second, the PDF file format is text-based, and text-based parsers are frequently subject to bugs in their source code that lead to code execution vulnerabilities.

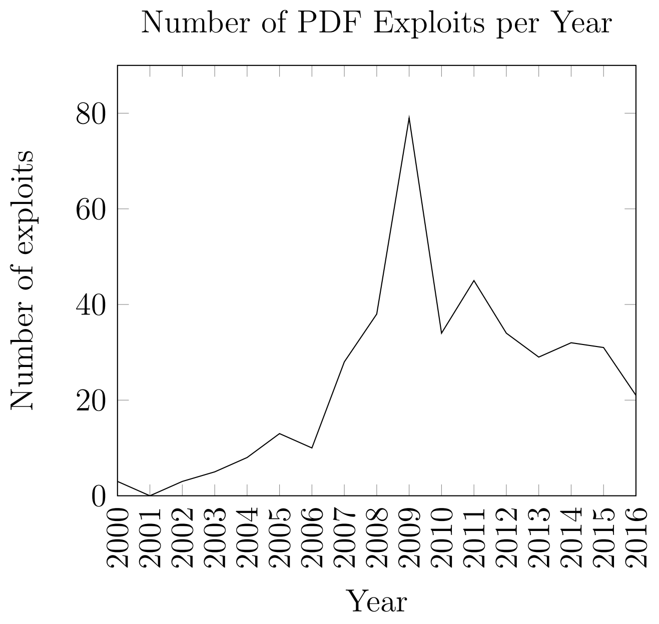


Figure 1.7. The trend in the number of PDF vulnerabilities over time, independent of product.

Figure 1.7 above shows the trend of reported PDF vulnerabilities according to MITRE's CVE database (MITRE, 2016). These data are for vulnerabilities disclosed prior to July, 2016. Also note that the exploits do not belong to any one particular product such as Adobe Reader, but rather belong to a variety of different pieces of PDF rendering software. A total of 411 different PDF vulnerabilities were reported.

Unique PDF vulnerabilities peak in 2009. What changed in 2009 was that Adobe instituted a regular, quarterly patch cycle followed by improvements in the security model used for the Adobe Reader software package (Adobe Product Security Incident Response Team, 2012). After the improvements of 2009 the current trend is for PDF vulnerabilities to maintain a steady average around 30 per year with 2016 on track to meet or exceed that number.

1.4 Ransomware

Ransomware is a subclass of malware with a particular method of operation. After a computer is infected by ransomware the program begins encrypting different types of files like documents and spreadsheets. The user is then unable to access their own files. The ransomware alerts the user to the compromise and provides a series of steps necessary to decrypt the affected files (Microsoft Malware Protection Center, 2016).

Phishing attacks are a popular vector for delivering ransomware. Phishing data collected by PhishMe for the month of March 2016 that shows over half the malware taken from phishing emails is ransomware (2016).

One solution to being infected with ransomware is simple: pay the ransom. Hollywood Presbyterian Medical Center succeeded with this approach. They paid their ransomers 40 bitcoins (around $17,000 at the time of writing[1]) and received a cryptographic key for use in decrypting their ransomed files (Winton, 2016).

But paying the ransom is no guarantee of relief. Because ransomware is a crime so there is no contract in place to enforce that the ransomer will in fact follow through on his/her of the bargain. This is exactly what happened to another hospital, Kansas Heart Hospital in Wichita. They paid their ransom only to have the criminals turn around and demand even more cash (Sun, 2016).

1.5 Research Question

Phishing continues to be a serious problem for the information security community. Research into the phishing problem is ongoing and discussed in more detail in the following chapter. But much of this research looks at the language of

[1]Bitcoin's value relative to other currencies is highly volatile.

the phishing email as a secondary indicator, or somehow less important than the unambiguous technical details that are provided via the SMTP protocol. And those research projects that do look into phishing language stay at the shallow end of natural language processing. There is a lack of research into the semantic meaning behind phishing emails. Having insight into the structured meaning underpinning these phishing emails could provide better, more flexible solutions in fighting them.

There are two parts to the research question for this dissertation. The first part is: What would the semantic representation of known good emails and phishing emails look like? And second: Do the semantics differ enough between the two to enable semantic analysis and distinction? The end goal for the second part of the research question would be a system that could automatically distinguish between good and phishing emails based on their semantic parses. But building a complete system to do this is outside the scope of this dissertation.

1.6 Delimitations

Delimitations, as opposed to limitations, are explicit decisions on the part of the researcher that restrict the scope of the research.

One important delimitation in this research is the focus on phishing emails with malicious PDF attachments. The reason for this delimitation is two-fold. First, PDF files capable of exploiting a vulnerability in the software on a user's machine do not occur at random. The odds of such an occurrence are statistically impossible due to the sheer number of complementary changes needed to a valid file. The attaching of a malicious PDF file is there a guaranteed sign that the goal of the email is to make a phishing victim out of the end user. Second, PDF files are primarily text documents. Focusing on a particular type of file means limiting the number of action verbs related to that file type. A user doesn't "execute" or "run" a PDF the way one would a PE32 .exe binary. Or a user wouldn't "listen to" or "play" a PDF in the same way as an MP3 audio file. Since the goal of this dissertation is to search for and identify triggers in the text that would cause a user to complete the phishing attack, it seems prudent to limit the number of variables involved. In this case, one of the variables is the set of all possible action verbs.

1.7 Limitations

Limitations are present in any research endeavor. Understanding and explaining the limitations of the chosen approaches is necessary in supporting the arguments that naturally follow. A researcher who cannot explain the model or theory used may have built the entire research program on faulty assumptions and bad logic.

There are limitations inherent with using a knowledge-base approach such as that of ontological semantics. The expressive power of the ontology and lexicon may prove insufficient to describe all possible text found in phishing emails. No one individual has enough time or energy to define all the necessary resources. That commitment is certainly outside the time frame of any dissertation.

The proposed solution to this limitation is to create concepts and lexemes, from the ontology and lexicon respectively, on the fly as the email messages are analyzed. This makes the assumption that the way these resources are defined is both coherent and valid within the scope of ontological semantics. Such an approach would make for an inefficient way to acquire resources in the long run due to the additional work necessary to scaffold together all the disparate pieces.

1.8 Summary

This introduction chapter presented the basic concept of phishing emails. It then established why phishing emails continue to be an important problem in the field of information security. After establishing the severity of the problem, the proposed dissertation research is described in regards to its goals, limitations, and delimitations. The following chapters continue to build on and expand on these themes.

CHAPTER 2. REVIEW OF RELEVANT LITERATURE

Like all men of the Library, I have traveled in my youth; I have wandered in search of a book, perhaps the catalogue of catalogues...

Jorge Luis Borges
"The Library of Babel"

This chapter addresses the review of literature relevant to the dissertation. The literature review covers several different topics due to the different solutions to phishing that were proposed over time. The goal of this chapter is then to provide a narrative for these previous works and explain their importance to the original work described later.

2.1 Information Security

This dissertation addresses a problem central to information security. The past decade has seen an increase in the general awareness regarding information security problems. Dr. Matt Bishop of the University of California, Davis authored the first authoritative and comprehensive textbook on information security (Bishop, 2002). Bishop's text covers a wide range of topics from information security to cryptography and malware. Once the theoretical basis of information was established the next logical question was: What real world information security questions might be answered via the experimental method. Long after Bishop, Dr. Josiah Dykstra wrote the text on how information/cyber security might become an experimental science (Dykstra, 2016). Dykstra's book begins with the scientific method and then describes different scenarios for its application.

One way of understanding security is to begin by understanding the attackers. In 2011, Lockheed Martin produced a paper where they borrowed the concept of the "kill chain" from the military and applied it to cyber defense (Hutchins, Cloppert, & Amin, 2011). The original military idea of a kill chain was the set of procedures necessary to identify and destroy a time-sensitive target as embodied by the unwieldy acronym, F2T2EA ("Find, Fix, Track, Target, Engage, Assess," Tirpak, 2000).

The cyber kill chain identifies the steps of a cyber attack in terms of increasing level of severity beginning at target reconnaissance and ending with a hacker remotely interacting with the target machine. Each step in the cyber kill chain has different indicators of activity and as such has different preventative and remediative measures available.

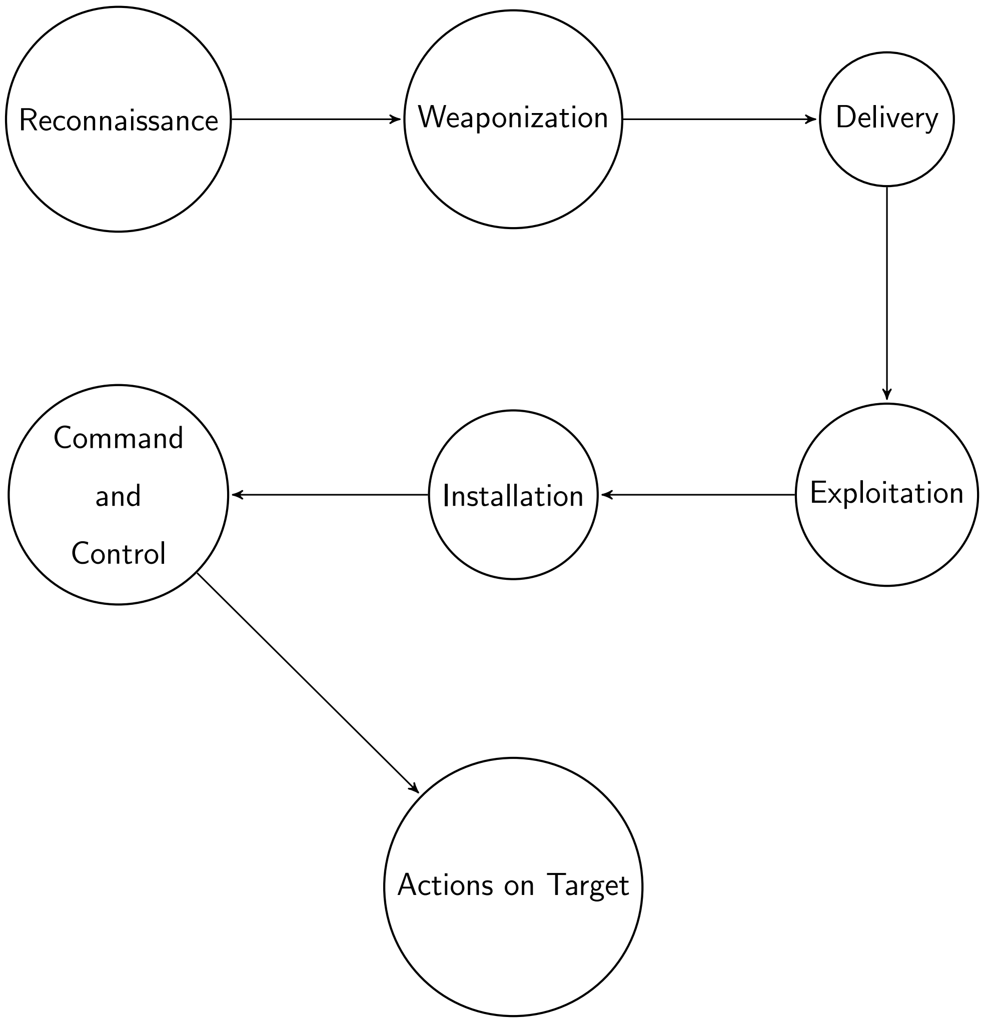


Figure 2.1. The Lockheed Martin cyber kill chain model.

Phishing attacks would sit at step three of the cyber kill chain, delivery. The phisher has already identified targets and prepared an attack. Now all that needs to happen for the attack to succeed is for the victim to perform some action. If the potential victim does not perform that action then the attack fails and all further steps in the cyber kill chain are prevented. Prevention of this attack can be accomplished by education and awareness training and/or technical preventative measures that either keep the phishing email from reaching the intended recipient or prevent the attack portion from succeeding despite the victim performing the attacker's desired action (Kumaraguru, Sheng, Acquisti, Cranor, & Hong, 2010).

2.1.1 Social Engineering

Kevin Mitnick was a successful computer criminal until his arrest (Shimomura, 1996). He has since parlayed his high profile court case into a security consulting and speaking career. Many of Mitnick's successes as a hacker came not from superior technical tools or know-how, but rather from social engineering the necessary information for his attacks from the victims themselves.

Social engineering itself is a modern euphemism for a confidence scam. In fact, when Mitnick wrote a book on social engineering with several case studies on different attacks he concluded each attack with a section headlined, "Analyzing the Con." (Mitnick & Simon, 2002) Chapter seven of that book is titled, "Phony Sites and Dangerous Attachments," and addresses specifically the social engineering involved with phishing attacks.

Even creating phishing emails for the attack phase can be done using automated tools. SpearPhisher from TrustedSec (2013) is intended for use by information security professionals who conduct penetration testing as a part of their regular responsibilities. Metasploit, the popular crowd-sourced exploit framework produced by Rapid7 (2012) also facilitates the creation of phishing emails. What these tools have in common is their ability to incorporate a template email message with a list of name-email address tuples and a malicious payload. The tool generates all desired email with a minimum of interaction on the part of the phisher. This removes the skill requirement for phishing attacks, lowering the barrier to entry, and enabling more attackers.

2.1.2 Software Exploits

Computer software can contain flaws that are otherwise known as “bugs” in computer science parlance. The more complex the software product the more likely it is to contain bugs (Mayer, 2012). When such a bug is exposed to the outside world it creates a vulnerability that threatens not just the individual piece of software in question but the system that software is running on and perhaps the entire network that that system belongs to. Exploits are pieces of software specially crafted to use one or more vulnerabilities in order to manipulate a computer system into functioning in an unintended way (Koziol et al., 2004).

Not all vulnerabilities are the same. Some vulnerabilities leak information used by the vulnerable software or stored elsewhere on the system (Tsipenyuk, Chess, & McGraw, 2005). Other vulnerabilities cause the software to crash or become unresponsive when exploited. These kinds of bugs can create denials of service where system availability is compromised. But the most severe type of vulnerability (Microsoft Security TechCenter, 2012) is the kind that when exploited allows the attacker to execute arbitrary instructions on the system in a kind of “weird machine state” (Dullian, 2011). These remote execution vulnerabilities give hackers from distant networks potentially unrestricted access to the system and all of its resources (Open Web Application Security Project, 2013).

2.1.3 Exploit Kits

Exploit kits are just one type of tool used by phishers (Segura, 2015). The kits collect together software exploits that target vulnerabilities in web browsers or other web browser components. Examples of popular exploit kits include Angler, Nuclear, Magnitude, and Neutrino (J. C. Chen & Li, 2015; Howard, 2012). The phisher establishes a publicly accessible web server somewhere on the Internet. Then attack emails are sent out with links back to the exploit kit server in the hopes that users follow the link using a vulnerable browser and get their computer exploited. This is an effective way to deliver malicious PDF attachments such as those being examined in this research (Stock, Livshits, & Zorn, 2015).

Sometimes these kits include OPSEC features to help protect the phisher's investment in infrastructure and increase the number of successful attacks. For example, the malicious link to the exploit kit server might embed a unique identifier. The first time the server receives a connection request using that ID it will attempt an exploit. But subsequent attempts using a previously seen ID will simply serve innocuous content and attempt no exploitation. This way, if the link is forwarded to a security professional for in-depth analysis, it is less likely that the professional can trigger the server into openly demonstrating its malicious behavior.

2.1.4 Related Attacks

Examiners of phishing have subdivided this class of cyber attack into smaller pieces over the years based on differing sets of criteria. One subclass of phishing attack that has reached the public media is that of spear phishing.

Spear fishing, the original and properly spelled term, differs from more common rod and reel fishing in that the fisherman must pierce a single fish with a lengthy spear. To do that the fisherman must first identify and target a particular fish. The angler doesn't necessarily target a specific fish but rather casts the bait out and waits for any fish at all to take the bait and hook themselves. Spear phishing when compared to the more general type of phishing attack is analogous to the real world distinction between the two types of fishing. Whereas a phisher sends large amounts of emails with the hopes of catching a few victims, the spear phisher targets a single particular target and crafts the message of the phishing attack to that one target.

Whaling is spear phishing with a narrower focus. The targets of whaling, the "whales", are determined by the attacker to have a higher value than the other users, which in this analogy are common fish to be ignored. CEOs, politicians, or celebrities are all potential whales.

On the African savanna water is a scarce and valuable resource. Seasonal water holes offer temporary sources of water to any animal fortunate enough to find them. Predators know this fact as well. Lions, alligators, and other apex predators stake out these water holes, waiting for prey animals to feel the pang of thirst and wander into striking distance. Water hole cyber attacks are analogous to the kind of attacks that these African predators execute. A water hole attack begins when the attacker identifies a web site or other Internet-based resource frequented by many users. The attack then compromises the site via vulnerabilities in the host system and inserts his/her own malicious content into what was originally hosted. Now all the attacker has to do is wait for unsuspecting users to visit the compromised site and fall prey to the software exploit waiting for them.

2.2 Knowledge-Based Theory

When beginning to research a new field it is often helpful to have a definitive text of that field to use as a starting point. For knowledge representation the foundational text is Sowa (2000). John Sowa presents a large and diverse number of theories in the realm of knowledge representation and discusses each in turn as to its expressive power and other attributes.

2.3 Machine Learning

Machine learning is the field of computer science that applies algorithms to data sets in order to automatically find patterns that are not obvious to humans either due to the complexity of the pattern or the large amount of data involved.

ML itself is considered a subfield of artificial intelligence (AI) where the goal is the computational generation of models. The basic theory and foundation of AI is established well by Russell and Norvig in *Artificial Intelligence: A Modern*

Approach (2003). The broad range of material presented in the book, along with the mixture of theoretical grounding combined with practical examples, make this text the book of choice for many university artificial intelligence courses.[1]

The learning part of machine learning is grouped into three general categories: supervised, unsupervised, and semi-supervised.

Unsupervised Training data is provided as-is without any labels.

Supervised All training data is labeled.

Semi-supervised A combination of unsupervised and supervised training methods. Often use when there is a large amount of data or labeling data is expensive or time-consuming.

Supervised learning approaches typically perform better than unsupervised approaches for the same data/problem. This fits the intuition that being able to better describe your input data produces more accurate results when the trained model is applied to test data. But sometimes a complete structure of the data is unknown. Or a sufficient labeled corpus doesn't exist. In these situation unsupervised ML can provide an answer.

Data mining is another subfield of AI and is closely related to ML. The key different between data mining and ML is that data mining prioritizes the search aspects examining data while ML takes the techniques and tools of data mining to utilize large amounts of data. Together the two subfields work synergistically. Witten, Eibe, and Hall establishes the ideas and techniques of data mining and machine learning (2011). If the authors' names sound familiar it would be because they are three of the creaters of the WEKA machine learning toolkit that will prove of central importance to this research (Hall et al., 2009).

Different ML algorithms perform better on particular kinds of features. Investigating this in regards to semantic features is part of the focus of this

[1]UC Berkeley maintains a list of universities that use this book on their web site at: http://aima.cs.berkeley.edu/adoptions.html

research. An alternative to a single ML technique that performs well on all types of features is combining different ML techniques together in order to derive the best aspects of the different parts. Combined ML algorithms together is known as ensemble machine learning.

There exist a wide variety of ensemble ML techniques. In a voting ensemble approach the component ML algorithms all get a “vote” in the resulting classifier. Random forests build a single classifier from a collection of decision trees each trained on a random subset of samples (Breiman, 2001). There is also boosting, a technique that focuses on taking a collection of “weak” classifiers and assembling them together into a single, “strong” classifier (Breiman, 1998). These are just a sample of the various ensemble techniques currently available.

2.4 Phishing Detection

Phishing detection is an area of ongoing research. No definitive solution has been found to date. This is likely due to the fact that phishing attacks succeed or fail on their ability to interact with a real human, and processing and analyzing text in a way similar to how humans perform those tasks remains an unsolved problem. Still, there are a few different avenues being pursued.

First addressed is general machine learning (ML). There are several useful contributions that ML-based phishing detection approaches make.

The next area is natural language processing (NLP). Many current NLP approaches use ML techniques. But while the preceding ML section focuses on solutions that use features derived from email meta data, NLP approaches extend the research into looking for useful feature derived from email message bodies.

2.4.1 Features

Sometimes the features chosen for ML aren’t just linguistic but also include meta data. Meta data would be information not explicitly stated in the content but rather gained by examining the artifact in which the text is presented. CANTINA+ (Xiang, Hong, Rose, & Cranor, 2011), and its preceding version, CANTINA Yue, Hong, and Cranor (2007), rely on these kinds of meta data features.

The goal of CANTINA+ is to identify phishing web pages. These are web pages that mimic existing web pages of companies such as banks and email providers. An unsuspecting user is directed to the phishing page via an email prompt to log in. The phishing web page interceptions the user login credentials, allowing the phisher to log in as the victim. Because phishing web pages are often scraped versions of legitimate web pages, using content-based features won't distinguish them from their good counterparts. CANTINA+ focuses on meta data features such as:

1. Domain - Aspects of the URL used for the web page such as the number of subdomains, if an IP address is used in the URL, or embedding other domain names as a part of the larger phishing domain name.

2. HTML - The relative location of validation scripts for the web page form or when the anchor text mismatches with the corresponding tag's URL.

3. Web - How long ago the domain name was registered, Google PageRank score, or TF-IDF scores for the suspect page versus the page given as the top search result for the same brand name.

The first two types of features, domain and HTML, are fairly static. That is to say, evaluating a given web page against these features will not generate different feature scores over different periods of time. However, the third type of feature, web, is dynamic sensitive to changes because it has external dependencies on web-based services such as search. Running a search to retrieve scores for the features will likely generate different results over time. This makes replicating the research results difficult and possibly complicates generalizing the results.

Another interesting point to note about the evolution from CANTINA to CANTINA+ is that in CANTINA the researchers wrote their own metric by hand.

They could do this because the number of features they were using was relatively small. When dealing with small numbers of features it is possible for a human to manually write a metric or heuristic that performs competitively. But with the CANTINA+ project the number of features used grew, as suggested by the article title. When working on CANTINA+ the researchers decided that they needed to use ML techniques to develop classifiers using this large assortment of features on hand. Their work utilized naive Bayes, logistic regression, and SVM to train classifiers among other algorithms.

The two CANTINA projects are frequently cited by other phishing research efforts. But their focus is on identifying web pages used to steal user credentials in phising attacks. Refer back to the model depicted in Figure 1.5 to see how this type of phishing fits in the bigger picture of phishing attacks.

CANTINA+ inspired a number of different follow-on research projects to include PILFER (Fette, Sadeh, & Tomasic, 2007), phishGILLNET (Ramathan & Wechsler, 2012), and PhishAri (Aggarwal, Rajadesingan, & Kumaraguru, 2012). Where PhishAri differs from PILFER and phishGILLNET is its focus on phishing via the web site, Twitter, instead of the more general task of identifying phishing web pages.

2.4.2 Natural Language Processing

Language poses an additional set of problems for ML approaches. Years of linguistic research has shown different types of structure present in language. For ML to perform NLP well it needs to needs to account for these structures via some kind of model.

Noted linguist Dan Jurafsky writes extensively on NLP. His text, *Speech and Language Processing*, serves as an excellent bridge from the Russell and Norvig's more general, *Artificial Intelligence: A Modern Approach.*

Ontological semantics, the language-centric knowledge-base approach to NLP utilized in this research, is not a newcomer to the problems of information security. Theoretical foundations exist for a number of information assurance and security problems such as watermarking, anonymization, and document sanitization (Raskin, Hempelmann, Triezenberg, & Nirenburg, 2001; Raskin, Nirenburg, Atallah, Hempelmann, & Triezenberg, 2002; Raskin & Taylor, 2013).

Just as there are several levels of evaluation used in linguistics, so too are there several comparable levels of features that an NLP solution might utilize.

2.4.2.1. Surface Features

ML approaches for NLP often begin with what are known as surface features. These are strings of text extracted from the target corpus. The attraction to surface features is that they're very easy to extract in an automated fashion with a minimum of prior training or supervision.

N-grams are one of the most commonly used types of surface features where the gram is just a word. The N parameter defines how many words concatenated together should be considered as a single feature. A unigram model (or 1-gram) would create a feature for each and every word in the text. A bigram model (or 2-gram) creates a feature for each possible two word sequence. For example, the text, "How now brown cow," would generate four unigram features or three bigram features. Usually trigrams are the longest size of gram chosen due to the fact that increasingly longer grams failed to provide improved results.

Some experiments also combine different sizes of N. Both unigram and bigram features might be considered together if they are shown to improve the results produced by the model.

N-gram language models can create an explosion of features. A larger number of features slows processing and potentially decreases the accuracy of the

Table 2.1

An example of different length N-grams.

	"How now brown cow"
Unigrams	"How", "now", "brown", "cow"
Bigrams	"How now", "now brown", "brown cow"
Trigrams	"How now brown", "now brown cow"

model. To help lower the number of features and remove features of the language model that contribute little to the accuracy, ML practitioners turn to morphological analysis and filtering.

Morphological analysis is one processing technique that can help cut down on the number of grams generated. Linguists describe written words according to basic units called morphemes. Many languages allow changes and additions to these basic morphemes in order to derive new meaning without having to create an entirely new word. Reversing these additions in order to obtain the core morpheme of the gram is how morphological analysis works. Analysis techniques differ from language to language and require a scientific level of linguistic research in order to develop.

In addition to morphological analysis, ML techniques might also filter out what are known as stop words as a pre-processing technique. These are common words in a language, often belonging to closed classes of words such as articles and prepositions. Stop words usually serve a functional purpose and stand in contrast to content words from which most meaning is derived.

Aggarwal, Kumar, and Sundarsan (2014) improve on their prior work on the PhishAri project (Aggarwal et al., 2012), by examining linguistic features in addition to the meta data features previously used. The approach is largely enumerative:

1. Monetary terms and symbols

2. Words like “reply” and related synonyms
3. Time-sensitive words and synonyms thereof (urgency)

The ability to use synonyms comes from relying on the WordNet database. WordNet is a large collection of English words with connections between synonyms (Princeton University, 2010). Its popularity among researchers comes from its free cost, its large coverage of words within the English language, and the associated linguistic meta data about connections between words. These connections are often used in NLP approaches that advocate semantic levels of processing.

2.4.2.2. Syntactic Features

Gilchan Park of Purdue University took a step beyond linguistic surface features (Park, 2013). Stop word filtering. Park’s work was inspired by another project at Purdue University, PhishNet (Prakash, Kumar, Kompella, & Gupta, 2010). The focus on the research is identifying salient action verbs. A part of speech (POS) tagger is applied to the token stream to identify which words are most likely the verbs. These potential verbs are checked against a like of action verbs and their synonyms.

Park himself didn’t stop with his masters thesis. A 2014 paper conducted an empirical study comparing both machine learning classifiers to humans assigned with the same task (Park, Taylor, Stuart, & Raskin, 2014). Human responses were collected via Amazon’s Mechanical Turk micro crowdsourcing application. This study showed an advantage on the part of machines to outperform untrained users on the phishing detection task.

Park and Taylor also took Park’s earlier work and developed it into more detail (Park & Taylor, 2015). This later research focused on the various ways in which syntactic phrase structures might generate machine learning features. Syntactic connections between the set of action verbs, and their subjects and objects. This is another small step away from NLP surface features into meaningful, semantic features.

2.4.2.3. Semantic Features

Semantics is a less frequently examined topic in phishing research. Vector semantics (Salton, Wong, & Yang, 1975) appears most often when studying ML-based semantic NLP. Both the previously mentioned phishGILLNET (Ramathan & Wechsler, 2012) project and other work such as that of L'Huillier, Hevia, Weber, and Ros (2010) utilizes latent semantic analysis as the primary ML tool.

But this kind of semantics is fundamentally different from knowledge-base forms of semantics. Latent semantic analysis and latent Direchlet allocation both take input texts and cluster the individual words according to what their latent meaning or semantics. The space in which these word clusters exist expresses the hidden or latent semantics. But discovering what exactly is the specific meaning behind these clusters still remains a task for humans.

The WordNet project out of Princeton University offers another approach to semantics. WordNet is just as its name suggests; a network of words (Princeton University, 2010). The network is made up by describing how the words are interrelated via hypernyms, hyponyms, synonyms, and other lexical features. These connections are something of a projection of the underlying semantics but they are still lexical and language-specific. For instance, Arabic has single words that distinguish between maternal and paternal aunts and uncle. English makes no such distinction so the hypernymic/hyponymic relations between the words in the two languages would necessarily differ.

Meaning-based machine learning (MBML) is a research program outlined by Falk and Stuart to improve traditional machine learning by infusing it with actual, human meaning (Falk & Stuart, 2016). The hypothesis behind MBML is that only by providing semantically meaningful data as input to a ML system will the output be meaningful. This stands in direct contrast to vectors in semantic space or networks of lexemes.

2.5 Summary

This chapter identified the existing literature that is relevant to the dissertation. There are several different, related fields at work in this research.

Table 2.2

A list of anti-phishing academic research projects.

Solution	Source	Media	Classifier(s)
AntiPhish	2005	Web page	Custom heuristic
"Approximate string"	2014	URL	Custom heuristic
CANTINA	2007	Web page	Custom heuristic
CANTINA+	2011	Web page	SVM, logistic regression, Bayesian network, J48, random forest, AdaBoost
"Distributed software agents"	2013	Web page	Multi-agent system
"Experimental comparison"	2000	Email	Naive Bayes
"Genetic algorithm"	2010	Web page	Genetic algorithms
GoldPhish	2010	Web page	Custom heuristic
LinkGuard	2006	URL	Custom heuristic
PILFER	2007	Email	SVM, rule-based, decision tree, Bayesian
PhishAri	2012	Twitter	Naive Bayes, decision trees, random forest
PhishCatch	2009	Email	Custom heuristic
phishGILLNET	2012	Email	Probabilistic latent semantic analysis, AdaBoost
PhishNet	2010	URL	Custom heuristic
Phishpin	2009	Web page	Custom heuristic
PhishTackle	2014	Web page	SVM, customer heuristic
Phishwish	2009	Email	Custom heuristic
PhishZoo	2011	Web page	Custom heuristic
SpoofGuard	2004	Web page	Custom heuristic
Web Wallet	2006	Web page	Custom heuristic

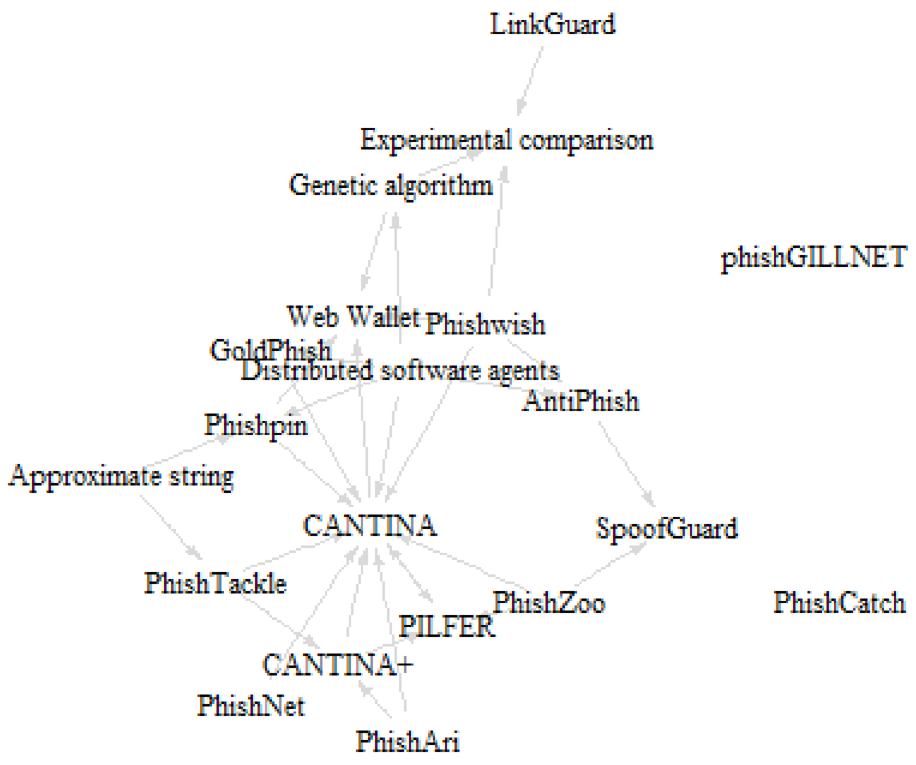


Figure 2.2. Viewing the citations of the references from Table 2.2 as a network.

Half of the foundation of this dissertation is information security. In the last ten to twenty years there were written several foundational texts on the topic. Newer, more specific topics have arisen since then such as complex web exploit kits.

If information security is half the foundation of this dissertation then knowledge representation is the other half. Knowledge representation concerns itself with the representation, storage, and retrieval of information stored in forms similar to how humans represent knowledge to themselves. These theories of knowledge representation play a key role in modern technologies such as natural language processing and the Semantic Web.

Machine learning is a suite of tools and techniques designed to find patterns in large amounts of data. ML originally grew out of the wider field of artificial intelligence. There are several different ways to compare ML techniques such as whether or not the input data is tagged or the ways the algorithm represent the pattern internally. ML plays a key role in this dissertation by providing the tools for comparing differing sets of data, and understanding how ML works is the first step towards meeting that goal.

The final field addressed by the literature review is that of phishing detection. Phishing attacks are not new, and they are a fairly well-studied field by this point. There exists research covering all the different varieties of phishing attacks, how to train users to recognize them, and the impacts that successful attacks have on the economy. Because this dissertation specifically focuses on using ML for identifying phishing emails it is useful to know the prior research in this area and what approaches were already tried.

CHAPTER 3. THEORY

In theory, there is no difference between theory and practice. But in practice, there is.

Yogi Berra

This chapter describes the relevant theories that underpin the experiments to follow. These theories are developed into formalisms used to transform input text into meaningful output structures. For an OST-based system the end output of processing is known as a text meaning representation (TMR). The TMR is a graph structure that represents the totality of information presented in the input corpus. A gold standard TMR would be one that is as close as possible to the understanding that a speaker would take away from reading it.

Ontological Semantics Technology (OST) is a frame-based system for processing natural language text (Taylor, 2010). OST recognizes the ambiguity inherent in natural language and doesn't constrain itself to being processable by first-order logic the way that pure ontology systems like the Semantic Web does (Berners-Lee, Hendler, & Lassila, 2001). Recanati makes a parallel distinction with "linguistic semantics" and "real semantics" where the former is necessary for handling the difficulties of natural language before passing the results on to the latter (Recanati, 2006). Work in the form of the OntoSem2OWL project looks to build a one-way bridge from the kind of ontology practice by OST and the kind of ontology practiced by the Semantic Web (Java et al., 2007).

One of the goals of OST is to use microtheories as a way to build its knowledge representation and processing capabilities out to meet the need of natural language. While the descriptions of linguistic phenomena in this paper do not meet the criteria of a formal microtheory they do provide a starting point for reasoning and describing aspects of the natural world.

3.1 Text Meaning Representation

The TMR represents the totality of knowledge available in a text. The facts, instantiated from abstract concepts, connect to one another via their slots.

In OntoSem, a RELATION is a subclass of PROPERTY where both domain and range are concepts. Properties with non-concept ranges would fall under the ATTRIBUTE property. Another feature of a RELATION that distinguishes it from an ATTRIBUTE is the ability for it to have its own INVERSE property. INVERSE is itself a type of RELATION. When a RELATION is connected to another RELATION via the INVERSE property then the domain of the first RELATION becomes the range of the second, and the range of the first RELATION becomes the domain of the second. This essentially creates a bidirectional link between the domain and range facts.

The sentence in Example 1 serves as a way of illustrating the high-level stages that text goes through in order to become a TMR.

(1) "Bob put down the book that he was reading."

Through a series of processing steps the resulting S-expression might look something as follows in Figure 3.1:

Figure 3.1 shows how concepts gleaned from the example form a nested structure. The same structure might also be represented as a tree as shown in Figure 3.3. This tree is an intermediate step in the formation of a TMR. But what the tree shows, that the TMR lacks, is the structure with which the information was originally presented in the text. In many ways this concept tree parallels syntax trees. For comparison purposes, Figure 3.1 shows one possible interpretation of a phrase structure.

```
(PUT–DOWN–22
    (AGENT (VALUE (HUMAN–41)))
    (THEME (VALUE (BOOK–3
        (PRINTS–CONTENT (VALUE (BOOK–CONTENT–7)))
        (THEME–OF (VALUE (READ–189
            (AGENT (VALUE (HUMAN–41)))
        )))
    )))
)
```

Figure 3.1. An intermediate TMR represented as an S-expression.

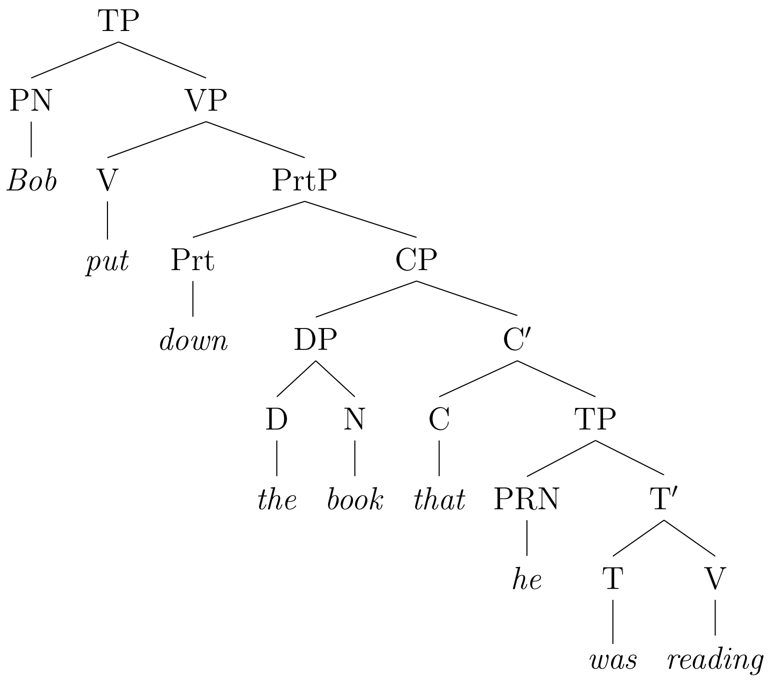


Figure 3.2. Phrase structure representation of Example 1.

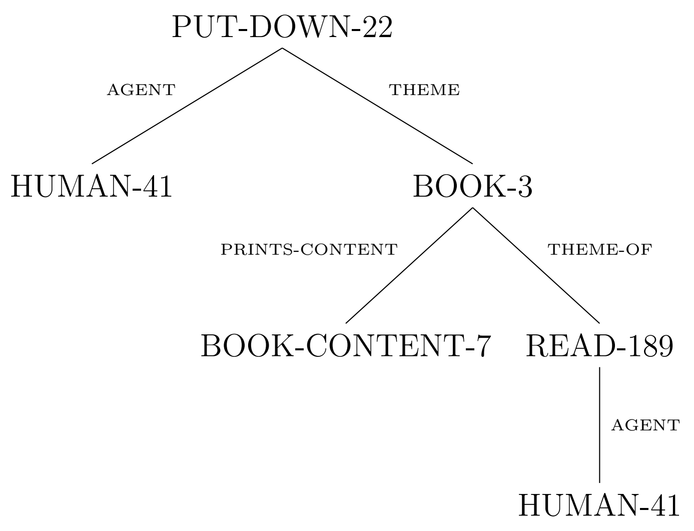


Figure 3.3. The tree structure of Figure 3.1.

There is a key piece of information about the ontology that is necessary to know before the Figure 3.3 concept tree makes sense. While the example sentence contains a single word, “book”, this could have two possible renderings according to the ontology (Guarino & Welty, 2002b). First, there is the physical, printed book in which strings of characters are written in ink on paper and bound together. Second, there is the content being presented in the book which instead of being physical is treated as a mental object, intangible and invisible. These two different concepts are connected by a relation that explains how the mental book content is attached to the physical book. What happens in the example is that both concepts are invoked in one utterance. In the main clause, “Bob put down the book”, the book in question is physical because it is the THEME of a physical event, PUT-DOWN. But in the relative clause, “the book that he was reading”, the meaning is now understood to be the mental BOOK-CONTENT object because it is the THEME of the mental event, READ. Therefore, the resulting concept tree must instantiate both of these two concepts and connect them together with the aforementioned PRINTS-CONTENT relation.

This concept tree only shows one direction of the relations between concepts. The TMR is a graph that includes all explicit information like that presented in the concept tree, and the implicit information contained in the INVERSE relations. For instance, the concept tree in Figure 3.3 states that BOOK-3 is connected to BOOK-CONTENT-7 via the PRINTS-CONTENT relation. But it is also true, yet not explicitly stated, that BOOK-CONTENT-7 is also connected to BOOK-3 via the INVERSE relation of PRINTS-CONTENT, IN-PRINTED-CONTENT.

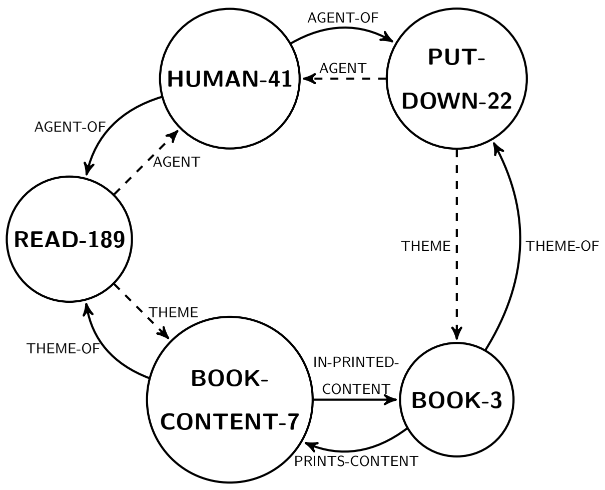


Figure 3.4. The complete TMR derived from the intermediate TMR in Figure 3.3.

3.2 Modeling Linguistic Phenomena

There exist several linguistic phenomena that warrant consideration when generating a TMR. The most important phenomena are structural in nature. Surface phenomena such as orthography and morphology aren't addressed in this paper.

Any automated solution for generating TMRs should address each of these phenomena. This paper doesn't devote space to discussing what such a solution might look like. So no assumptions are made about how the interactions between phenomena might translate into algorithmic implementations.

3.2.1 Syntax

Syntax is arguable the most difficult topic to cover. This is due in part to the gray area called the syntax-semantics interface; it is not clear where the area defined by syntax ends and the area defines by semantics begins, and vice versa.

The approach described in this paper looks at dependency syntax. This aligns well with the existing SYN-STRUC formalism (Nirenburg & Raskin, 2004). Note that this doesn't necessarily the dependency approach empirically better. It is just easier to immediately use as a SYN-STRUC doesn't describe the constituency need for a universal grammar approach.

3.2.1.1. Plurality

Plurality is a deceptively simple syntactic phenomenon. One concept instantiates into one fact. But how does a group of two instances of the same concept get represented?

The original OntoSem work provides a formalism useful in instantiating plural concepts. The SET structure offers a variety of parameters that combined can represent the variety of plural forms found in languages around the world (Nirenburg & Raskin, 2004).

(2) "A car."

A noun with a numeral is the simplest case of English plural. The SET's TYPE parameter would be assigned the noun's CONCEPT. In this situation "car" has a SEM-STRUC with the AUTOMOBILE concept. The CARDINALITY parameter is assigned the numeric value of the numeral, which is three. And finally

```
(AUTOMOBILE−98)
```

Figure 3.5. A trivially simple intermediate TMR with a single instantiated concept.

the COMPLETE parameter is assigned the TRUE value because Example 3 is describing a finite set of cars.

(3) "Three cars."

```
(SET−117
    (TYPE (AUTOMOBILE))
    (CARDINALITY (3))
    (COMPLETE (TRUE))
)
```

Figure 3.6. A set that describes three cars instantiated at once.

(4) "Cars."

3.2.1.2. Case Roles

Case roles highlight one of the problems involved in defining the syntax-semantics interface. Some syntactic approaches utilize features to mark constraints on the objects of a verb. But these features only hint at the underlying semantics. Below are the formally recognized case roles used by OST systems (Nirenburg & Raskin, 2004):

Table 3.1

The list of case roles as used by Ontological Semantics.

Case Role	Definition
AGENT	Entity responsible for an action.
BENEFICIARY	Entity deriving benefit from an action.
DESTINATION	Ending point of an action.
INSTRUMENT	Object used to carry out an action.
LOCATION	Place where an action occurs.
MANNER	Style in which an action is done.
PATH	Route taken during an action.
PATIENT	Entity affected by an action.
SOURCE	Starting point of an action.
THEME	Entity manipulated by an action.

```
(SET−203
    (TYPE (AUTOMOBILE))
    (CARDINALITY (>1))
    (COMPLETE (FALSE))
)
```

Figure 3.7. A set describing one or more car instances.

Work from Dik defines a hierarchy of semantic roles available to syntactic subjects. By default the subject of a sentence fills the AGENT slot of an event as shown in Figure 3.8:

(5) “Bob sent the spam.”

```
(SEND−11
    (AGENT (VALUE (HUMAN−73)))
    (THEME (VALUE
    (SPAM−EMAIL−MESSAGE−2)))
)
```

Figure 3.8. An intermediate TMR demonstrating a subject with agency filling an AGENT role.

Case roles don’t necessarily map one-to-one with their syntactic positions. It is possible that the subject of a sentence may actually be the instrument. This can be determined by examining whether or not the subject has the necessary agency to carry out the action in question.

(6) “The malware sent the spam.”

```
(SEND-12
    (AGENT (VALUE (MALWARE-793)))
    (THEME (VALUE
    (SPAM-EMAIL-MESSAGE-3)))
)
```

Figure 3.9. An intermediate TMR demonstrating a subject without agency filling an INSTRUMENT role.

3.2.1.3. Anaphora and Ellipsis

Anaphora is a deceptive syntactic phenomenon. Understanding anaphora requires resolving pro-form words to the entities they reference. There can be reflexive pronouns serving as direct objects that refer back in the sentence to the subject. Or the subject might be a dummy pronoun whose only purpose is to make the sentence syntactically correct: "It is hot out today."

Along with anaphora comes the phenomenon of ellipsis (McShane & Babkin, 2015). Ellipsis differs from anaphora in that ellipsis is the eliding, or deletion, of words or entire phrases of text. This is often seen in question answering. "Did you walk the dog today?" "I did [walk the dog today]."

Without proper handling of ellipsis the intermediate TMR of the response might look as follows:

This is wrong for a few reasons. First and foremost, it omits both the object of the sentence ("the dog") and the adverb that modifies when the event occurred ("today"). Secondly, instantiating the verb "did" as EVENT creates an overly vague TMR. We know from the preceding utterance in the discourse that "did" refers to the act of walking. A better intermediate TMR, one in which the available information is used to resolve the ellipsis would look like:

```
(EVENT−99
    (AGENT (VALUE (HUMAN−1023)))
)
```

Figure 3.10. How the phrase, "I did," might look without resolving the ellipsis.

```
(WALK−PET−4
    (AGENT (VALUE (HUMAN−1023)))
    (PATIENT (VALUE (DOG−74)))
    (TIME (''2015−08−15''))
)
```

Figure 3.11. The phrase, "I did," with the ellipsis properly resolved.

3.2.2 Semantics

The core methodology described in this dissertation derives from the work into ontological semantics (OntoSem) by Nirenburg and Raskin (2004). OntoSem uses a frame-based ontology with inheritance rules for structuring its knowledge. The ontology itself is language-independent. But each language gets its own lexicon whose lexeme entries map back to and optionally extend the concepts specified in the ontology.

One of the earliest OntoSem projects was the Mikrokosmos project at New Mexico State University (Onyshkevich & Nirenburg, 1995). Mikrokosmos was designed to support machine translation between English and Spanish. While not without its faults (Guarino & Welty, 2002a), Mikrokosmos established several key aspects that would feature prominently in later OntoSem work.

The latest iteration of OntoSem is known as Ontological Semantics Technology or OST (Taylor, 2010). OST continues on with the OntoSem program and looks to improve on what components and theories were deemed lacking in the prior versions.

Semantics is at the heart of this paper. Semantic information, in comparison to pragmatics, is information that may be determined in absence of any kind of context.

At the core of OST's knowledge representation is the ontology. Concepts are organized in a hierarchical fashion with child concepts inheriting properties from their parent concept (Touretzky, 1986). These seemingly simple IS-A relations hide potentially complex situations (Brachman, 1983).

3.2.2.1. Modality

Linguistic modality is similar to the modality found in first-order logic systems in that it modifies what would otherwise be a standalone expression. The sentence, "We eat Indian food," is a simple declaration. Add in the modal auxiliary verb "could" and the meaning changes to a statement of possibility, "We could eat Indian food."

OST defines a system of modalities consisting of seven different modal qualifiers, some of which are familiar to first-order logic systems as described in Table 3.2.

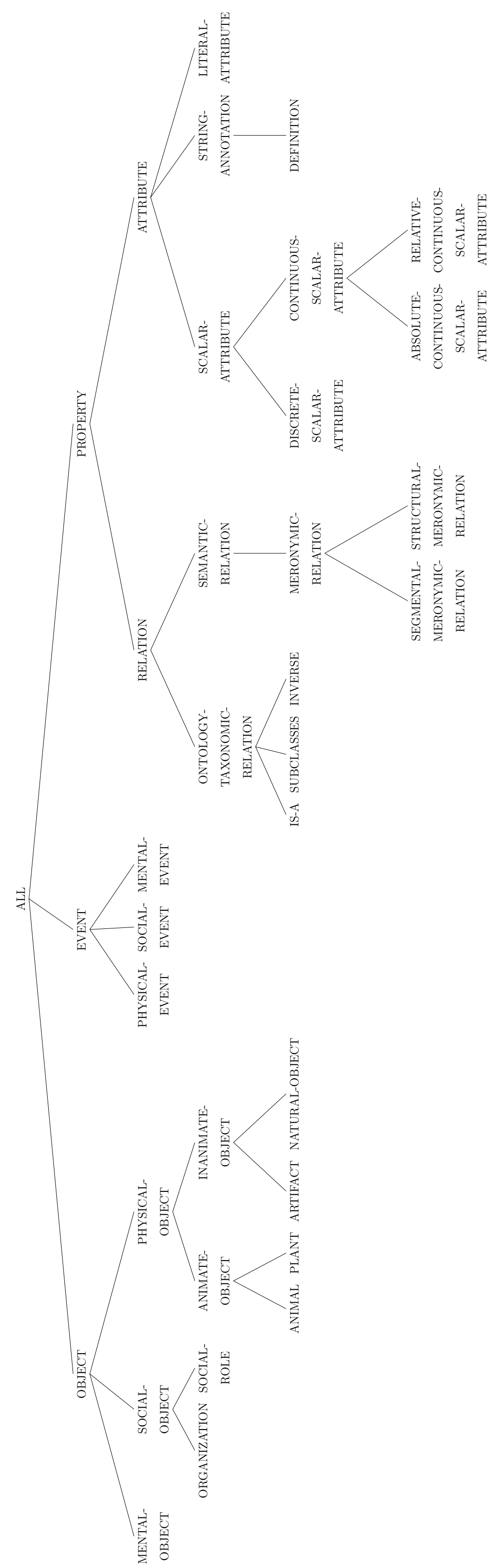


Figure 3.12. An example upper ontology.

Table 3.2
The list of modalities as used by Ontological Semantics.

Modality	Definition
Epistemic	Belief about the factualness.
Deontic	Moral obligation and permission.
Volitive	Degree of desirability.
Potential	Ability to perform the action.
Epiteuctic	Resulting success.
Evaluative	Attitudes expressed.
Saliency	Importance of the meaning.

There exist several problems with the current OST modalities not the least of which is a formally blessed way of expressing negation. Negation could be expressed as an epistemic modality with a value of 0.0. It seems more likely that negation is a unary boolean modifier and not a continuously valued modal qualifier. Another problem is that the deontic qualifier collapses and oversimplifies a distinction between obligation and permission.

3.2.2.2. Social Roles

Social roles are an interesting semantic phenomenon. A concept can play a social role without subclassing a new concept (Masolo et al., 2004). They are completely different from the previously discussed case roles.

Take a student as an example. First and foremost, a student is a HUMAN concept. If there were a separate STUDENT concept that inherited from HUMAN then there would be the situation that when the human-as-a-student were referred to it might be as the STUDENT-1 while when the very same individual were spoken of on his/her own it would be as the HUMAN-2 concept. The fact that there are two separate and distinct facts suggest two distinct entities when this is directly contrary to the information being conveyed.

(7) "The president is sworn into office."

```
(SWEAR-IN-332
    (AGENT (VALUE (PRESIDENT-4)))
    (THEME (VALUE (POLITICAL-OFFICE-97)))
)
```

Figure 3.13. Instantiating a role instead of the individual playing the role.

```
(SWEAR-IN-332
    (AGENT (VALUE (HUMAN-59
        (PLAYS-ROLE (VALUE (PRESIDENT-4)))
    )))
    (THEME (VALUE (POLITICAL-OFFICE-97)))
)
```

Figure 3.14. Instantiating a role instead of the individual playing the role.

Instead, the student should be anchored with a HUMAN concept and use a PLAYS-ROLE slot to connect it to a STUDENT concept. In the preceding example, STUDENT inherited from HUMAN. But the improved way to handle it would be for the STUDENT concept to inherit from a dedicated SOCIAL-ROLE concept that in turn is a subclass of SOCIAL-OBJECT (as opposed to the PHYSICAL-OBJECT that is a HUMAN).

3.2.2.3. Complex Events

Complex events are given a brief treatment in the Nirenburg and Raskin text (Nirenburg & Raskin, 2004). A complex event has a mereological relationship with one or more other events that form parts of the larger complex event (Fellbaum, 2002).

McDonough establishes some criteria for what distinguishes a complex event from a simple, or atomic event (C. J. McDonough, 2000, p.32). An event should be considered as a complex event if its frame specifies:

1. Sub-events or a mereonymic nature (as opposed to a taxonomic subclass).

2. Preconditions that must occur before the given event would happen.

3. Postconditions or results that come from the event's completion.

Taking a step is one example of a complex event. There are three distinct sub-events that make up a STEP event: 1) the foot is raised, 2) the foot is repositioned, and 3) the foot is then set down. It should be noted that it is necessary for it to be the same foot throughout all three sub-events otherwise the STEP concept does not make sense. Without this additional constraint it would be perfectly valid for the STEP complex event to involve three different feet; one each for each sub-event.

Events are at the core of OST. When something happens, or an action is performed, it is an event. Many events are atomic, which is to say that they are self-contained and don't require describing any other component events. Contrast these simple, atomic events with complex events, whose definition requires other events exist in order to make sense of the larger, complex event. These part events may themselves be other complex events.

3.2.2.4. Scripts

The idea of scripts grew out of cognitive psychology research aimed at bridging the gap with artificial intelligence (Schank & Abelson, 1977). Continued development of scripts was done by Schank and Leake under the title of memory organization packets Schank (1999).

Scripts are defined in comparison to events. Whereas an event is self-contained and atomic, a script may contain several interacting components and multiple different pathways to completion. Another distinction is that an event can stand in isolation, but a script is only understood after studying multiple occurrences of a phenomenon.

As an example, compare the event SHINE with the script ATTEND-PLAY. SHINE is the physical emission of light. It's instrument could be a LIGHT-BULB or a CANDLE or even a CHEM-LIGHT. The external product is LIGHT.

The script for ATTEND-PLAY is fraught with more complications than the SHINE event. For ATTEND-PLAY there may be the necessary precondition of PURCHASE-TICKET. Only then can the HUMAN with the TICKET corresponding to the DRAMATIC-PLAY in question enter the PERFORMANCE-VENUE, find the SEAT assigned to them by the TICKET, SIT, and then WATCH with rapt attention. Additional paths arise if we consider that the play may be a free production and not require a ticket. Or the seating is general admission so an assigned seat doesn't exist. Somehow there must be a logical path through the ATTEND-PLAY script that invokes one or more of these components events.

Scripts are related to complex events, but differ in significant ways. While all scripts are complex events, not all complex events are scripts. Scripts differ from complex events in two important ways:

1. A script is developed from repeated observations of common sequences of events as they happen in the real world. Complex events can be described and understood in a vacuum devoid of the context needed by scripts.

2. The component events of a script are causally connected. One component event happens that flows into the next and so on. Complex events, on the other hands, require all their component events to happen.

As an example of a complex event, take respiration, the process by which many living objects obtain the oxygen they require in order to live. Respiration is a complex event. Its part events are inhale, and exhale. Both inhalation and exhalation must happen and must happen only in the specified order for the complex respiration event to have occurred.

For a script example consider teaching. A treatment of the teach event is found in Nirenburg and Raskin (2004). But according to the two differences between complex events and scripts described above, teach would more accurately be described as a script. The reason is because while teach might include instruction on the part of a teacher and learning on the part of a student only the former is required for the teach event to occur. Teaching might result in learning, so the learn event follows causally from instruct, but learning is in reality only a condition for success of teaching. If no learning occurs then the teaching still occurred, but it was largely a failure.

Schank and Abelson (1977) offer the first formal treatment of both scripts and plans. Plans can be treated as a more general case of scripts, but won't play a part in this dissertation. The goal of Schank and Abelson (1977) was to bridge cognitive psychological research with that of computer science. This theory section will expand to include a description of how scripts might fit into an OST system along with a formalism useful for describing any scripts encountered in the methodology chapter.

So how do atomic events, complex event, and scripts differ? As a rule of thumb, an atomic event can always be a complex event (C. McDonough, 2000). It is always possible to describe an event in more detail. For instance, describing walking as a sequence of steps using alternating feet. And steps being a sequence of raising, moving, and then lowering the same foot. This is a situation where the level of granularity in describing events is determined by the needs of the application. A medical application might require an in-depth description of respiration while a banking application might not.

The real question then is: how does a complex event differ from a script? Recall that scripts are informed by repeated observations of common sequences of events. One question an acquirer must ask themselves is: which parts of this sequence might be abstracted away, and which parts are perhaps culture or situation-specific? In many ways a script as defined by Schank and Abelson is more pragmatic than purely semantic because it encodes details that aren't universally necessary or relevant. Think back to the teaching script example. Americans raised in the 1950s might say that teaching involves students sitting at individual desks with a teacher standing in front of a blackboard, lecturing them on a topic. This teaching script stands in stark contrast to some more modern trends in teaching espoused by teaching philosophies such as Montessori, Aurora, and other "constructivist" schools.

There are then two basic tests for determining if an action is a complex event or a script:

1. There are two or more paths that might be followed through the sequence of events to reach an outcome that closes the script.

2. There are aspects of the script that are culture-specific or that otherwise don't generalize to all situations (either geographically or temporally).

3.2.2.5. Metonymy

Metonymous speech is a common phenomenon where one proper noun substitutes for another. Places can substitute for people who work there such as when news reporting cite the executive mansion in place of the executive staff. “The White House said today...” (Nirenburg & Raskin, 2004) The BUILDING concept does not have the necessary agency to fill the AGENT slot of the SPEAK event. The result as shown in Figure 3.15 is nesting a BUILDING instance beneath the implied HUMAN instance, connecting the two by way of a WORKS-IN relation.

```
(SPEAK−71
    (AGENT (VALUE (HUMAN−202
        (WORKS−IN (VALUE (BUILDING−9)))
    )))
    (THEME (VALUE (...)))
)
```

Figure 3.15. Building standing in place of a worker.

Or an individual can stand in for their work. “I was reading Palahniuk earlier...” The second HUMAN, Palahniuk, cannot fill the THEME slot of the READ event. The resulting TMR shown in Figure 3.16 inverts the structure used in Figure 3.15.

George Lakoff writes extensively on the topic of metonymy as it relates to the wider field of metaphor (Lakoff, 1987). His list of English metonymous relationships remains a staple and a solid foundation for beginning new knowledge representation work:

1. PRODUCER standing in for PRODUCT - Nothing sounds as lovely as a *Stradivarius.*

```
(READ–23
    (AGENT (VALUE (HUMAN–310)))
    (THEME (VALUE (BOOK–47
        (AUTHORED–BY (VALUE (HUMAN–311)))
    )))
)
```

Figure 3.16. Author standing in place of their work.

2. OBJECT standing in for its USER - That *taxi* totally ignored me.
3. CONTROLLER standing in for something that is CONTROLLED - The *pilot* nearly crashed.
4. INSTITUTION standing in for the responsible PEOPLE - *Mossack Fonseca* helped a lot of people conceal their wealth.
5. PLACE standing in for the INSTITUTION - *Number 10 Downing Street* released a statement today.
6. PLACE standing in for the EVENT - There should never be another *Hiroshima.*

3.2.2.6. Mereology

Mereology is the study of parts and their relations to whole objects. For instance, a wheel can stand on its own as a distinct concept, but car requires the wheel part in order to be fully realized and understood. Having a proper mereological structure for OST is necessary to accurately describe both objects and events. It should be noted that this dissertation is more concerned with the latter.

One avenue of mereology research distinguishes between structural and segmental types of mereology (Pribbenow, 2002). Structural mereology looks at how objects interconnect as parts in a larger structure. Contract that with segmental mereology that aims to describe how a single, whole object might be segmented up into parts as a function of natural language discourse.

3.3 Summary

The theory chapter of this dissertation outlines the linguistic, philosophical, mathematical, and computational theories necessary to turn a piece of raw, unstructured text into its TMR form.

This chapter began by describing exactly what a TMR is. It then proposed a structure called an intermediate TMR that exists between the syntactic phrase structure of a text and the full, formal TMR. This intermediate TMR is critical to the subsequent machine learning experiments.

Finally, the chapter addressed a series of linguistic phenomena that potentially affect how a TMR represents the meaning of a text. The list of linguistic phenomena included here is by no means comprehensive, and bears some bias towards the English language.

CHAPTER 4. FRAMEWORK AND METHODOLOGY

Sometimes science is more art than science. A lot of people don't get that.

Rick Sanchez
Rick and Morty

The methodology is proposed and a limited set of samples is examined. 10 samples each from known good and phishing emails were identified and used as examples. All of these samples included at least one attachment that was a PDF document. In the case of the phishing emails these attachments were malicious and were handled in such a way as to avoid accidental infection of the computers used to conduct the research.

Ten samples of known good emails were taken from the personal email accounts of the author. Finding good email samples for analysis isn't difficult in and of itself. Corpora such as Spam Assassin and the 2005 TREC provide samples (Cormack & Lynam, 2005). The difficulty comes in first finding sample emails with their attachments included. More difficult was finding sample emails whose attachments were specifically PDF documents. The Enron corpus was another possibility (Cohen, 2015). But later changes to the published Enron corpus include stripping out email attachments, which was a critical requirement for the research question.

Finding phishing email corpora is also a complicated task. The APWG willingly shares gigabytes of data with researchers who inquire. But the data require filtering to remove text artifacts created when an APWG member forwards the suspected phishing email to the group. After filtering there is still the task of parsing the emails and identifying which, if any, include PDF documents as attachments.

Table 4.1

Available email corpora.

Corpus	Type	Attachments	Size	Cost
APWG	Phish	Yes	˜94,000/mo.	Free
Avocado	Ham	Yes	614,461	$1600
Clinton Emails	Ham	No	30,322	Free
ELRA-WC347	Phish	Unknown	210	Unknown
Enron	Ham	No	˜500,000	Free
SpamAssassin	Ham/Spam	Yes	6047	Free
TREC 2007	Ham/Spam	Yes	75,419	Free
W3C	Ham	Yes	174,311	Free
Wolverhampton	Spam	No	673	Free

The MIME::Parser module for Perl offers a simple and effective library for parsing emails while simultaneously extracting their attachments (Eryq & Skoll, 2015). Once all of this processing is complete, finding an email with a PDF attachment is as simple as searching the corpus directory structure for files with a .pdf extension. Alternatively, anti-virus tools like ClamAV can scan the directory structure and identify malicious files, identifying phishing emails with previously seen malware samples to which the anti-virus tool possesses a known signature (Kojm, 2013).

4.1 Sample Selection

The first step in analyzing emails is the collection of one or more email corpora. To train a binary classifier there must be two corpora; a corpus of known good emails and a corpus of phishing attack emails. Table 4.1 describes a number of email corpora found during the initial stages of research.

The Anti-Phishing Working Group (APWG) is an information security industry group working to raise awareness and find solutions to phishing attacks (Anti-Phishing Working Group, 2015). They have a steady stream of sample emails that are forwarded to them by individual partners. The emails are available for download free of charge after registering an account. Emails are grouped by the month in which they were submitted with an average of over ninety thousand (90,000) emails per month.

Avocado is the collected emails of an information technology company that went out of business (Oard, Webber, Kirsch, & Golitsynskiy, 2015). The corpus is available through the Linguistic Data Consortium (LDC) for a fee of $1600 if you are not a LDC member. Multiple inquiries around the Purdue West Lafayette campus failed to uncover anyone with an LDC membership that might be able to acquire the emails free of charge.

The ELRA-WC347 corpus proved difficult to access. ELRA-WC347 is advertised by the European Language Resources Association (ELRA). This particular corpus collects 210 phishing emails from years 2004 and 2005. But the corpus is not freely downloadable. Requests to ELRA for access went did little to clear up the way to license or otherwise access the data.

The Enron email corpus is possibly the most well known of all email corpora (Cohen, 2015). Following in the wake of a tumultuous bankruptcy and federal criminal proceedings, the Enron corpus collects the corporate email communications of the energy trading company before its failure. The corpus has undergone several iterations with filtering for PII.

Hillary Clinton is the focus of an ongoing federal investigation regarding her use of a personal email server for handling work emails during her tenure as U.S. Secretary of State (O'Harrow, 2016). Some of these emails were revealed to contain sensitive, confidential, or classified information. Several unclassified emails were released in response to a Freedom of Information Act (FOIA) request (U.S.

Department of State, 2014). They are available as PDFs on the U.S. State Department web site but don't include any attachments.

SpamAssassin is a spam email filtering software system. As a part of their work, the SpamAssassin team assembled a corpus of ham and spam emails (2004). The suggested uses of these emails are as training and testing aids, and they can be downloaded separate from the SpamAssassin software package.

The Text Retrieval Conference (TREC) is a series of conferences devoted to the testing and measuring of text analytics software. TREC 2007 included a research problem focusing on correctly identifying spam emails. To that end, the TREC organizes built a corpus collecting known good emails and spam emails. Researchers would use this common resource as the means to test and compare their current state-of-the-art approaches to machine learning.

The W3C corpus included in Table 4.1 was produced by Yejun Wu of the University of Maryland, College Park (2005). The original source of the emails were a set of email mailing lists maintained by the World Wide Web Consortium (W3C). Wu's purpose in cleaning and filtering the mailing lists was their use in the 2005 TREC Enterprise Track (TRECENT) competition. Wu wasn't the only researcher to clean and distribute a version of the corpus and other academics have similar collections also available online.

Students at Wolverhamption University in the United Kingdom entered the Language Resources and Evaluation Conference (LREC) 2002 (Krishnamurthy & Orasan, 2002). As a part of their research they collected around fifteen hundred spam emails, which filtered down to 673 unique emails. This spam email corpus is freely available online from their Wolverhamption web page.

For the phishing samples it was decided to use the APWG data. The reason is because the APWG corpus is open-ended, constantly collecting more data from its partners. Since the sample emails needed to contain PDF documents as attachments then it made sense to use a source that could be continually revisited until a sufficient sample size was obtained.

Forming the known good email sample set was more difficult. Phishing emails, like spam emails, are unsolicited and sent to a large number of recipients who are unknown to the sender. The sender can't then personalize each and every email. As a result these unsolicited emails have a certain business/impersonal style (Yus, 2011). Therefore it was deemed necessary to find known good sample emails that shared this impersonal writing style. One option was bulk emails such as those sent to mailing lists. A request went out to academics at Purdue for emails received from mailing lists with a PDF attachment. The responses comprised the known good corpus for the subsequent research.

4.2 Pre-processing

All emails arrived as individual files. Each file is an RFC 822-encoded email message (Crocker, 1982). To get just the message body itself these raw emails must be individually parsed and the separate parts extracted. This process also includes parsing out any and all attached files.

Several tools and libraries exist for a variety of programming languages to parse emails. The final decision was to use the MIME::Parser module for the Perl language (Eryq & Skoll, 2015). Perl was designed and implemented to be a fast text processing language, which matches well with the text-focused task of processing emails. Each email file was parsed into its own destination directory, including any and all attachments.

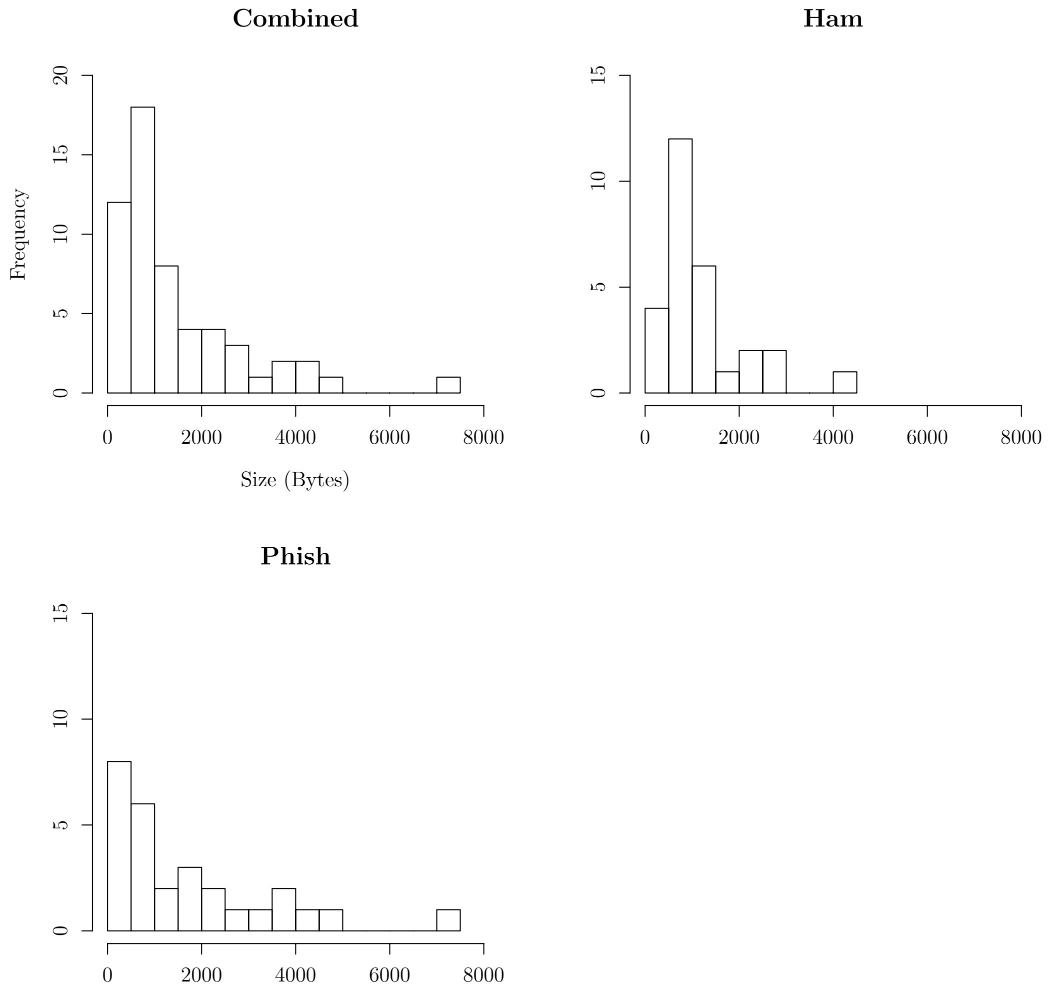


Figure 4.1. Histograms of the sample email size combined (upper-left), known good emails (right) and phishing emails (below).

Figure 4.1 shows the size distribution of the resulting sets of samples. The subsets of known good and phishing emails are broken out separately for comparison purposes. All three distributions show similar left skews.

4.3 Filtering

The APWG source for phishing emails provides far more samples than are needed. It's important therefore to filter and winnow down the raw corpus to a manageable size.

Several problems were encountered when examining the emails provided by APWG. Recall that APWG collects emails that are forwarded by participants. By default many email clients prepend forwarded text with some string like "> " or "| " to show that it is being cited by the email author. All this prepended text needed removing from any sample phishing emails.

Some APWG participants took the opportunity to annotate or editorialize the samples they were submitting. "Someone should let Microsoft know that their trademarks are being misused," suggested one participant. Other participants weren't even confident in their own submissions. Another asked, "Could someone tell me if this is legitimate?" These extraneous interjections, just like the prepended text inserted by the forwarding operations, must also be removed.

Another complication to the filtering task was that some APWG participants would forward multiple phishing emails, including them all as attachments in a single email sent to APWG. This caused problems because once the email file was run through the MIME::Parser module it wouldn't be apparent which extracted attachments corresponded to which phishing emails. Because the original research question focused on emails with PDF attachments, the inability to correctly connect attachments to messages meant that any phishing emails submitted in this way had to be discarded.

It is possible that the same phishing email was submitted by multiple participants to the APWG archive. Deduplicating, or removing duplicate instances of phishing emails, produces a set of unique samples. This smaller set of unique samples is quicker to analyze.

Deduplication was conducted by producing an MD5 hash of the message body for each email. Hashing just the message bodies removed many of the fields such as timestamps and destination email addresses that change with every email, making the entire email appear different to a cryptographic hash function such as MD5. But because MD5 is sensitive to minor input changes by design, it still wasn't a guaranteed way to provide 100% accurate deduplication. Even a single difference in whitespace between two email message bodies would cause their hashes to be different.

One of the original stated requirements of this research was focusing on emails with PDF documents as attachments. Once the MIME::Parser module did its work it was a simple matter of searching for files with a ".PDF" extension. The message bodies corresponding to these PDF files were collected together for final selection.

This research concerned itself only with English language texts. Cross-lingual or multi-lingual research is outside of the scope considered. So only English language emails would be used.

Each of the email message bodies corresponding to a PDF attachment were examined in turn. The language was noted, and the general theme of the message recorded. Identifying themes proved useful in the final deduplication phase. For instance, several emails that survived the prior deduplication step all used the same "Microsoft Online Sweepstakes" scam. Only one of these emails was needed in order to avoid over-representing emails of this theme.

During this language identification/deduplication step there was an interesting ancillary finding. The same theme and message appeared in three different languages: English, Spanish, and Portuguese. This suggests that there was an active phisher who took the time to produce the same attack script across multiple languages in order to reach a broader base. Whether or not the translations were done by a human or an automated service like Google Translate remains outside the scope of this research and was not pursued further.

4.3.1 Malware Detection

The final filtering step was scanning the candidate emails for malicious files. This step used the ClamAV tool (Kojm, 2013). ClamAV is a freely available, open source, signature-based anti-virus scanning platform. While ClamAV is usually deployed on servers to perform automated scanning it can also be run manually from the command line in the form of the clamscan tool.

None of the known good emails included malicious attachments. But four of the phishing emails did: 04, 05, 08, and 13. All four of these samples were extremely terse with none of them being larger than a kilobyte in size. It seems possible that phishing emails with malicious attachments are deliberately shorter than phishing emails that must elicit a response from a user. All the malicious attachment phishing email needs is for the user to open the attachment in question. To do that they provide a short, emotionally charged message meant to provoke the user into action quickly without thinking too long about the possible consequences.

4.3.2 Visualization

Word clouds are a useful tool for visualizing a large collection of text. They collect together the most frequently used words in a corpus with the most frequently occurring words being sized larger than the less frequently occurring words. This collection of differently sized words is then arranged both horizontally and vertically together in an ellipsoid shape and presented to the user. The word cloud offers at a glance a brief summary of the text statistics.

To visualize the text from the corpus at the end of the filtering stage, two word clouds were generated; one for the known good email bodies and the other for the phishing email bodies (Figures 4.3.2 and 4.3.2 respectively).

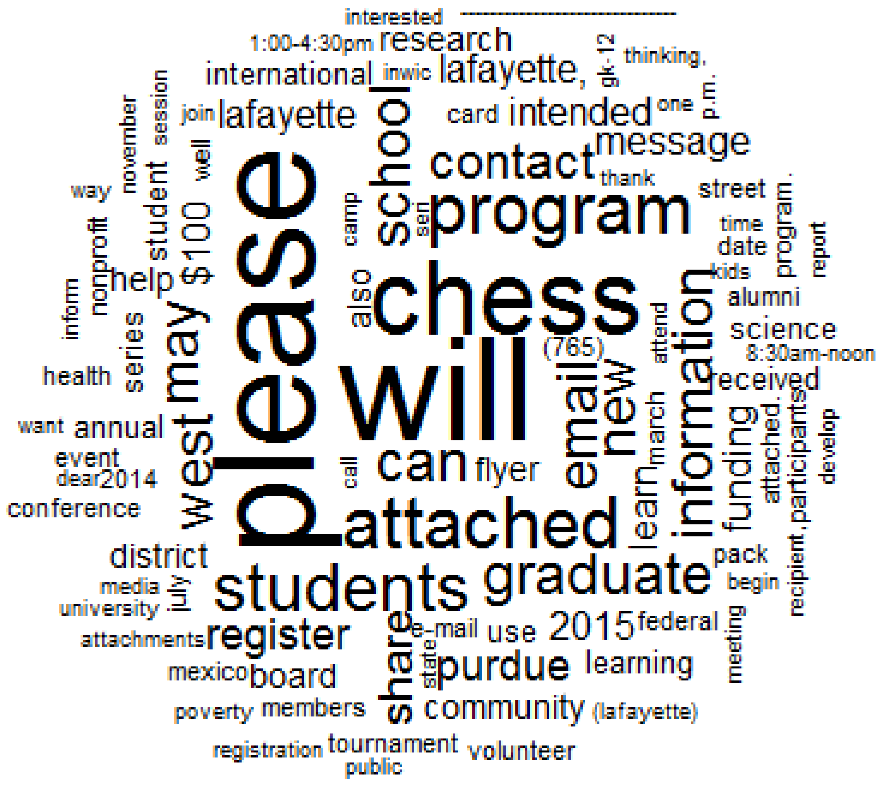


Figure 4.2. A word cloud of the one hundred most frequently encountered terms in the known good email set.

Figure 4.3.2 shows the word cloud for the known good (“ham”) email bodies. Already there are some interesting indicators appearing. While generic terms such as “please” and “will” appear largest, there are several academic-sounding terms that also appear such as “school,” “graduate” (whether this is a noun, adjective, or verb sense is unclear at this time), and “purdue.”

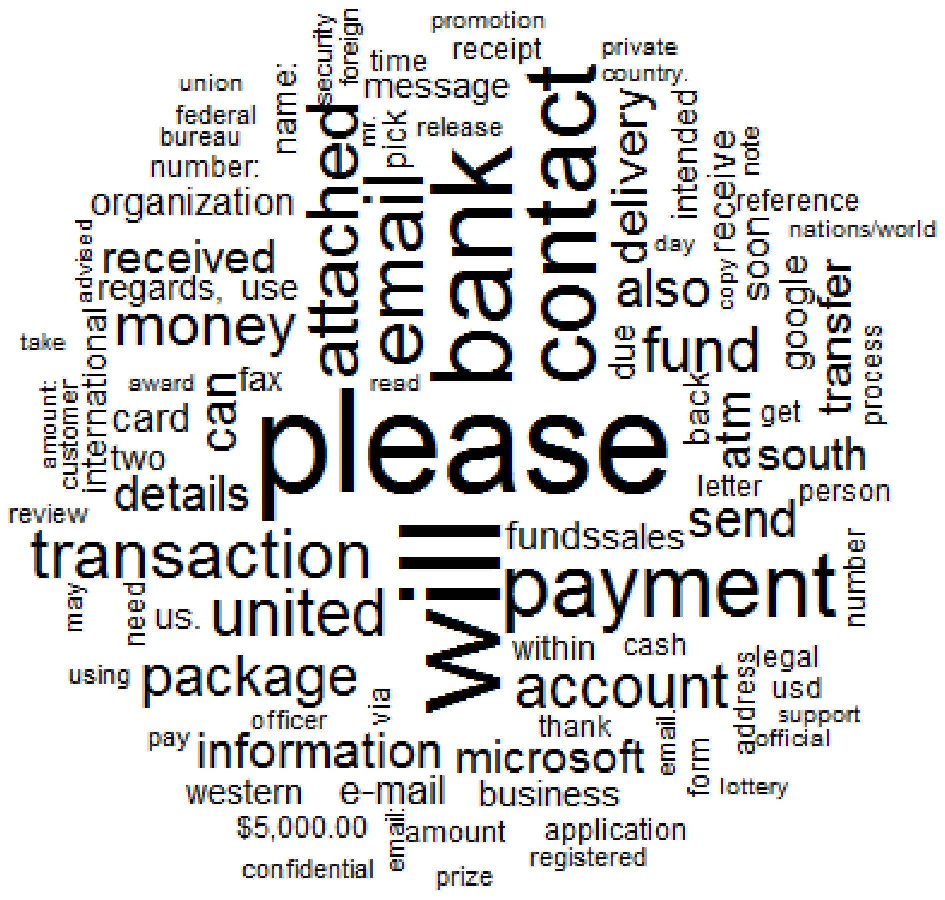


Figure 4.3. A word cloud of the one hundred most frequently encountered terms in the phishing email set.

Figure 4.3.2 shows the word cloud for the phishing email bodies. There are some superficial similarities to Figure 4.3.2 such as the terms “please” and “will” appearing prominently. But the smaller terms differ greatly. The phishing word cloud has several financial-sounding terms such as “transaction,” “bank,” and “money.”

4.4 Analysis

Each of the sample emails was analyzed in turn. One TMR was generated per email. The TMR describes the semantic information present in the email text without making assumptions about the implied intent. Linguistic phenomena such as anaphora and deixis are accounted for in the manual analysis. More complex situations such as those invoking scripts are called out in notes in the analysis without being explicitly marked in the TMR as such.

The analyses will appear superficially similar to the phrase structures present in syntactic approaches like universal grammar (Radford, 2009). This is largely coincident. While there exists evidence for connections between syntax and semantics it is not explored in this dissertation. Use the sentence, *The document is a PDF*, as an example. Figure 4.4 below shows an example tree of what the phrase structure might look like under universal grammar. Compare and contrast that with the OST-style semantic representation: (DOCUMENT-117 (HAS-DOCUMENT-TYPE (DOCUMENT-TYPE-43))) where DOCUMENT-TYPE-43 refers to the PDF standard. Gone are the articles because their semantic purpose was helping to understand how to instantiate the nouns. The copular *is* is also gone because it served a purely syntactic function joining the two noun clauses together, and serving no real semantic purpose.

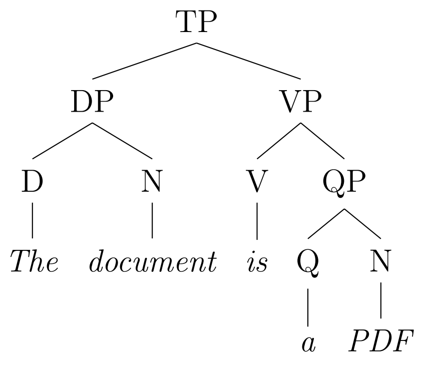


Figure 4.4. Phrase structure of the sentence, *The document is a PDF.*

There will not be an entire ontology and lexicon included as a part of the research. It will be assumed that concepts described are structured in the ontology according to best acquisition practices. Where doubt exists as to the correct placement of concepts then a note is included with more information as to the decision process.

There might be a need for recording the processing steps. Because the phishing emails might be identifiable by the amount and types of processing necessary when compared to ham.

4.4.1 Procedure

Ontological semantics distinguishes between an abstract concept and an instance of that concept seen in a text. An instantiated concept is known as a fact. The formalism used in ontological semantics is to provide the concept label in all capital letters. A fact is then labeled the same as the concept it instantiates but also includes a trailing hyphen followed by an ascending sequence number. As an example, HUMAN would be the label for the concept abstractly describing any human being while HUMAN-421 would be the 421st instance of the HUMAN concept that the system created.

Certain meta-data from the email, not present in the text of the message body, is needed in order to make sense of the rest of the email. The following data are used to instantiate concepts before the body analysis begins:

1. Speaker - The author of the email
2. Listener - The recipient of the email
3. Message - Every sample instantiates one email message and one or more attachments associated with that message
4. Timestamp - When the email was sent

Senders and recipients, or speakers and listeners, are always assumed to be two individual unknown to the analyst. This negates the need to cross-reference email addresses between sample emails and hides personally identifiable information.

Timestamps are crucial for disambiguating utterances such as “today” or “next week”. Without a specific anchor point for these references they will remain ambiguous.

The message bodies are processed by first breaking up the whole text into the individual sentences. Each sentence in turn gets a more in-depth level of processing. Every sentence generates one or more propositions, a structure anchored by an instantiated concept with zero or more properties coming off of it.

Table 4.2

Parameters changed from the default when using Weka's StringToWordVector class.

Parameter	**Value**
outputWordCounts	True
stemmer	LovinsStemmer
useStopList	True

The first changed parameter is the outputWordCounts. By default this is set to false, which means that instead of the numeric value of the attribute being the number of times the word occurs (the count) it is instead either 1.0 if the word is present or 0.0 if the word is absent.

Stemming is useful in statistical machine learning because it allows the computer to recognize when the same word happens in multiple inflected senses. For instance, "stemming," "stemmed," and "stem" are respectively the present participle, past participle, and simple present forms of the verb "to stem." It would therefore be a mistake to count each as a separate word with a count of one each. Instead, the "stem" word should have a count of three.

First of its kind, the Lovins stemmer described an algorithm for stemming English words (1968). This Lovins stemmer was chosen as the most basic approach to stemming words available in the Weka tool.

The third and final change to the default parameters was setting the useStopList to true. Stop words are strings ignored by the parsing algorithm. The motivation is to cut down on the number of total machine learning features by ignoring words like function words (prepositions, determiners, etc.), pronouns, and others. The actual stop list itself is the default option, the Weka−3−6 stop list.

4.5 Machine Learning

Training and testing of ML models was done with the Weka machine learning framework Hall et al. (2009). Weka is the state of the art tool for machine learning. It is written in Java and developed at the University of Waikato in New Zealand.

The ML experiments were conducted using three different algorithms. These three algorithms each represent a different category of ML classifier. Choosing multiple algorithms from differing categories provides the ability to examine classifier performance both across data sets and across specific algorithms. Weka

Table 4.3

Classifying the three different chosen algorithms.

	Linear	**Non-linear**
Generative	Naive Bayes	
Discriminative	SVM	J48

provides a variety of tools to include machine learning algorithms, visualization, and data pipelining for experiments.

Three different algorithms available in Weka were chosen for training and testing ML models. One algorithm each was chosen from different categories of algorithm to see how the two different data sets performed against different algorithms.

4.5.1 Naive Bayes

Naive Bayes is built upon the principles of Bayesian statistics. Recall that Bayes' theorem describes how the condition probability of an event based on previously observed conditions as shown in (4.1).

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{4.1}$$

The naive Bayes approach begins with the assumption of the conditional independence of prior events (Russell & Norvig, 2003). Naive Bayes works well despite this assumption, even for very small data sets such as those used in this research (Salperwyck & Lemaire, 2011).

John and Langley took this research and showed experimentally how it would function as a classifier (1995). It was this research that motivated the Weka implementation of the naive Bayes classifier. And because the classifier is probabilistic it assigns a continuous probability value to its results, letting a machine learning researcher examine how likely the algorithm things a certain class is.

X is the class of phishing emails

¬X is the class of non-phishing (known good) emails

w_i is a word present in the classifier

D is the document being examined

$$\ln \frac{P(X|D)}{P(\neg X|D)} = \ln \frac{P(X)}{P(\neg X)} + \sum_i \ln \frac{P(w_i|X)}{P(w_i|\neg X)} > 0 \tag{4.2}$$

(4.2) shows the math involved in the naive Bayes classifier. So long as the log-likelihood is greater than zero then the algorithm classifies document D as belonging to class X, phishing emails.

4.5.2 SVM

Support vector machines (SVM) are a popular choice of ML algorithm for NLP applications. For a binary classification problem, an SVM aims to draw a line that can cleanly divide the two classes (Cortes & Vapnik, 1995). Figure 4.5 demonstrates what this might look like in 2D space. All items on one side of the line belong to the first class of items and the items on the other side of the line belong to the other class.

The SVM implementation for Weka comes as an external package, LibSVM (Chang & Lin, 2011). Additional installation and configuration steps are necessary to get Weka and LibSVM inter-operating. Versions 3.7 and later of Weka feature package management software to facilitate the SVM installation.

SVM, unlike naive Bayes, is not probabilistic. All classifications done with SVM are either true or false with no probability assigned to the result.

(4.3) shows the math involved in a soft-margin SVM. Soft-margin SVMs are an extension of the basic SVM design. The changes allow an SVM to handle data

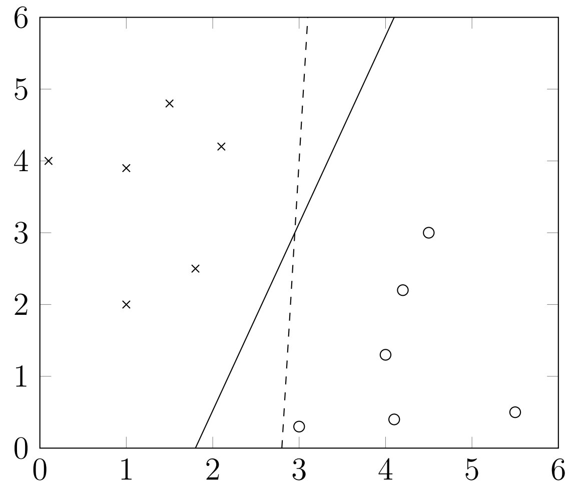


Figure 4.5. SVM training with a valid hyperplane (dashed) and an optimal hyperplane (solid).

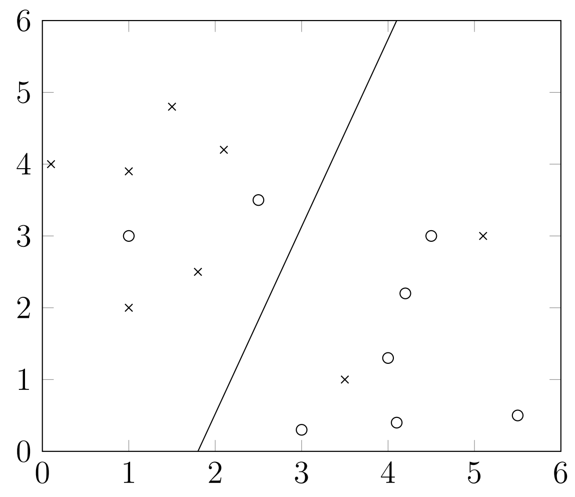


Figure 4.6. Data set where the two classes are not linearly separable.

where the classes are not linearly separable (it is not possible to draw a straight line/hyperplane to separate two classes such as the case shown in Figure 4.6).

$$\frac{1}{n}\ {}_{i}^{n}\ \max(0, 1 - y_i(\vec{w} \cdot \vec{x}_i - b)) + \lambda\|\vec{w}\|^2 \tag{4.3}$$

In (4.3), $\max(0, 1 - y_i(\vec{w} \cdot \vec{x}_i - b))$ is the hinge loss function that penalizes $\vec{x}$ data points that violate the $(\vec{w}, b)$ hyperplane. Increasing the λ parameter value up from zero allows more flexibility in terms of violations of the margin.

4.5.3 J48

J48 is a Java-based implementation of the C4.5 algorithm (Quinlan, 1993). Minor changes were necessary to avoid licensing issues involved with C4.5. In all other regards the J48 and C4.5 algorithms are identical.

J48 is a decision tree classifier algorithm. The ML features are used to build a decision tree model used for later testing and application. A benefit of the J48 algorithm over Naive Bayes and SVM is that the nodes in the decision tree contain

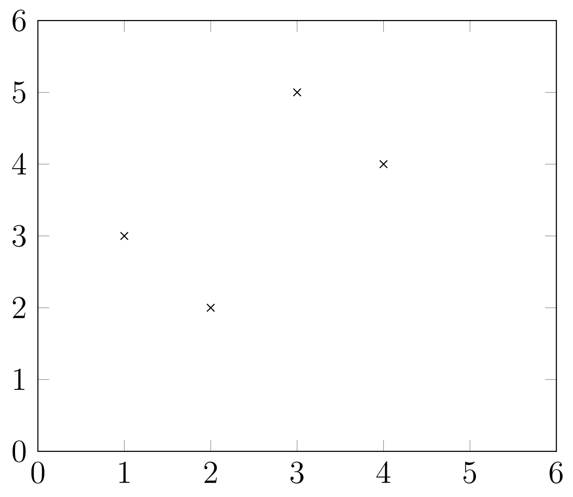


Figure 4.7. An example with four data points for training a machine learning regression model.

meaningful information as to the decision making process. More about this is discussed in the following sections.

It is also worth noting that John and Langley used C4.5 decision trees in their research into naive Bayes classifiers and found that sometimes naive Bayes did in fact perform better (1995).

4.5.4 Cross Validation

Cross validation is a general ML technique designed to prevent overfitting. Overfitting occurs in ML applications when a model fits the training data too closely and then produces poor results when used against general production data. A more desirable goal then is to train the model in such a way as to have a generally applicable result at the expense of the testing score.

Figure 4.7 plots four data points. For this example the goal is to train a regression model whose function generalizes well to production data.

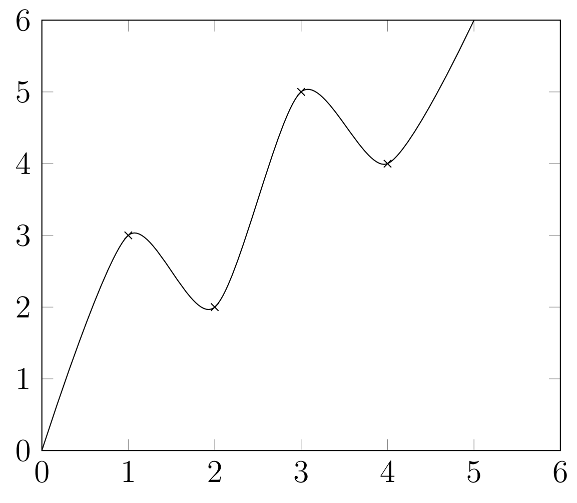


Figure 4.8. Machine learning regression model using a polynomial of degree five.

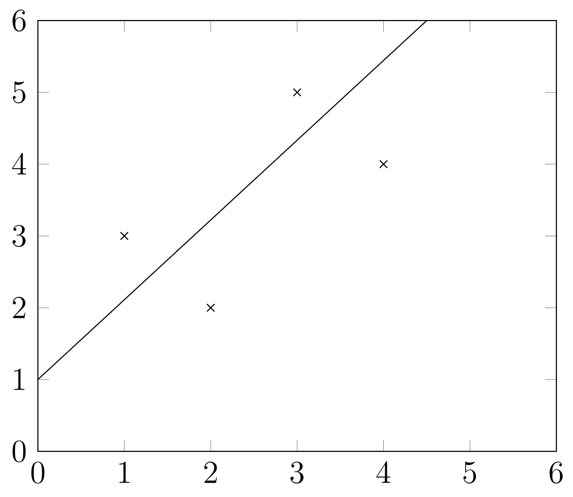


Figure 4.9. Machine learning linear regression model.

A polynomial function of degree five as seen in Figure 4.8 fits the data exactly. Accuracy is one hundred percent. This would seen to be an ideal solution according to the training data. However, the polynomial regression model is actually overfitting the training data. The problem is not apparent at this point.

The simpler linear regression shown in Figure 4.9 doesn't appear to fit the training data nearly as well as the polynomial in Figure 4.8. Accuracy is not one hundred percent and the function doesn't pass through any single data point. Technically the linear regression is underfitting the training data.

The problem of overfitting becomes apparent when taking a model trained on one set of data and applying it to novel data. Figure 4.10 shows the same polynomial regression model from Figure 4.8. None of the production data points are falling exactly on the line. But the one hundred percent accuracy seen in training would lead us to believe that this was a perfect model. Overfitting the data caused this model to not generalize well.

The linear regression model initially seen in Figure 4.9 is now applied to the same novel production data as Figure 4.10. Even though the training model didn't

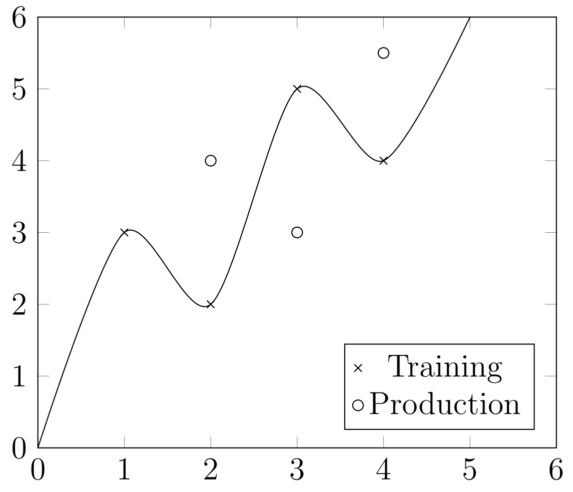


Figure 4.10. Machine learning polynomial regression model applied to production data.

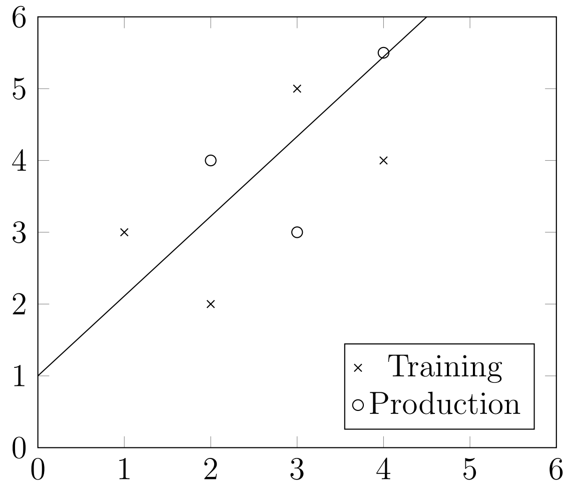


Figure 4.11. Machine learning linear regression model applied to production data.

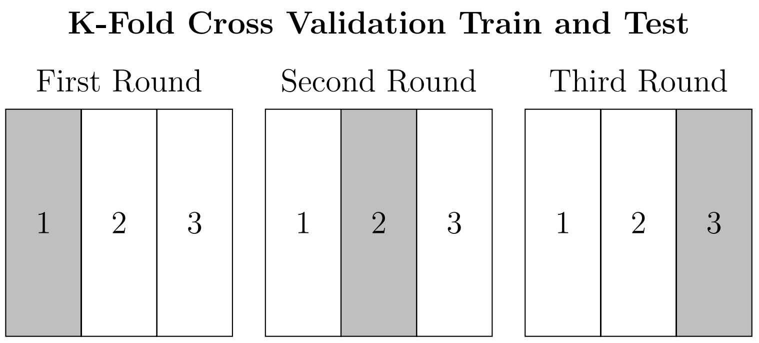
produce a perfect accuracy score it does appear to fit the production data roughly as well as it did the training data.

One approach to cross validation is called K-fold cross validation. In K-fold cross validation a parameter value is chosen, K, and the training data is evenly divided into K groups, or "folds." For each of the K rounds a different fold is help back for testing purposes while the other K-1 folds are used for training.

Table 4.4 shows how K-fold cross validation works when K=3. In this example, the grayed fold is the data fold used for testing in that round while the white folds are used for training.

Table 4.4

Illustrating K-fold cross validation training and testing with K=3.



For the experiments in this dissertation the parameter K was set to four. With twenty-eight samples in each of the two classes this meant that they could be divided evenly into folds of seven samples each.

4.6 Testing Hypotheses

After the machine learning classifiers are trained then the next task is to evaluate their performance. The unigram models serve as the baseline measure. They represent what is currently known to work for NLP solutions. The TMR models are the semantics-based experiments. Since there were three algorithms used to train both the unigram baseline and the TMR experimental models then there will also be three statistical hypotheses tested.

The statistical hypothesis test used for all three algorithms is the comparison of sample means where the sample means are the F_1 scores for each model. The F_1 score is used because it combines together both precision and recall into a single metric. These statistical tests determine, with a specified level of confidence, whether or not the results are meaningfully different.

4.7 Summary

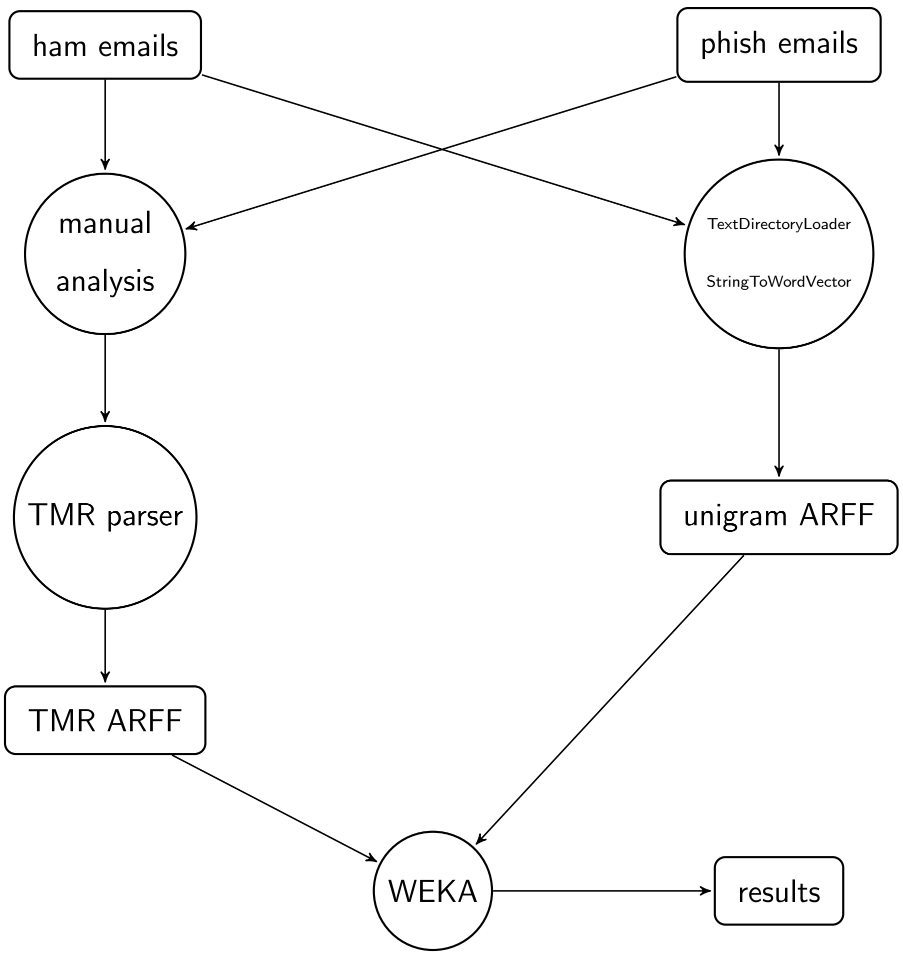


Figure 4.12. Data flow diagram of the proposed methodology.

The methodology of this chapter begins with the collection of email samples for the research process. It is important to note why certain types of emails were chosen or excluded, and the overall number of samples chosen. These emails underwent multiple filtering and processing steps before being passed on to later stages of the methodology.

The core research question in this dissertation is concerned with whether or not machine learning algorithms perform better when provided semantic structures as opposed to lexical n-grams. With a collection of samples assembled the next step was to analyze the text comprising the message body of each email and render its meaningful, semantic representation.

With the semantic data set in hand the third step was to train both baseline and experimental machine learning classifiers. The baseline classifier was trained on the uni-gram data set and the experimental classifier was trained on the semantic data set. Three different algorithms were trained on each of the two data sets to create a grand total of six different classifiers.

One final step remained, testing the performance of the machine learning classifiers to determine whether or not they successfully answered the research question. Statistical hypothesis tests comparing the sample means of the baseline and experimental data sets for each of the three algorithms.

CHAPTER 5. CONCLUSIONS

I am turned into a sort of machine for observing facts and grinding out conclusions.

Charles Darwin

The goal of this research was the identification of phishing emails using a knowledge-based approach to natural language processing. Phishing is a general class of computer attacks in which the attacker attempts to trick the target into completing the attack. Completing the attack may be as simple as opening an attached malicious file or opening a link to a malicious web page. Other times the attack might elicit personally identifiable information from the victim such as passwords or back accounts. The actual vector for the phishing attack is somewhat less important. Attackers might send phishing messages via email, social network posts, or cell phone text messages.

There is some overlap in the definitions of phishing and spamming. The key difference between the two is that the latter is done to the benefit of the phishing message sender (the attacker) and the latter is done for marketing purposes on behalf of some other third party. The spammer gets paid for sending messages in bulk regardless of the click-through rate (Krebs, 2014) while the phisher only benefits based on the small number of people who willingly fall victim to the phishing attack.

Overall this research was a success. It meets the original base goal set forth to produce a machine learning model that is trained on semantic data and performs better than random. The secondary goal of the semantic-based model performing better than its unigram-based counterpart is possibly met but due to sample sizes there can't be statistical conclusions drawn from the data.

5.1 Overall Results

There were six tests conducted in total. They were a cross between the two different data sets (unigram and TMR) and the three machine learning algorithms (naive Bayes, SVM, and J48). Table 5.1 below gives the empirical results for all six of these tests according to six different commonly used metrics: true positive rate (TP), false positive rate (FP), precision (Prec), recall (Rec), accuracy (F_1), and the area under the receiver operating characteristics curve (ROC).

Table 5.1

Results for all six tests scored against six different metrics.

Test	**TP**	**FP**	**Prec**	**Rec**	**F_1**	**ROC**
Uni-NB	0.800	0.404	0.673	0.800	0.723	0.741
Uni-SVM	0.754	<u>**0.207**</u>	<u>**0.813**</u>	0.754	0.762	**0.773**
Uni-J48	0.607	**0.271**	0.716	0.607	0.628	0.673
TMR-NB	**0.864**	**0.261**	**0.776**	**0.864**	<u>**0.813**</u>	<u>**0.905**</u>
TMR-SVM	**0.832**	0.286	**0.771**	**0.832**	**0.786**	**0.773**
TMR-J48	<u>**0.882**</u>	0.393	0.701	<u>**0.882**</u>	**0.773**	0.745

Table 5.1 calls out the results in two different ways. First, the top three tests in each metric are bolded. Second, the best test in each metric is also underlined in addition to being bolded.

TMR-based models rank higher in terms of the true positive rate and recall while the unigram model scores better in terms of loser false positive rates and improved precision. The relation between these measures makes sense after understanding how the precision measure incorporates the false positive rate while the recall measure incorporates the false negative rate.

Table 5.2

Confusion matrices for all six combinations of data sets and algorithms.

Uni-NB

		Classified	
		Ham	Phish
Actual	Ham	19	9
	Phish	11	17

Uni-SVM

		Classified	
		Ham	Phish
Actual	Ham	19	9
	Phish	5	23

Uni-J48

		Classified	
		Ham	Phish
Actual	Ham	17	11
	Phish	6	22

TMR-NB

		Classified	
		Ham	Phish
Actual	Ham	22	6
	Phish	8	20

TMR-SVM

		Classified	
		Ham	Phish
Actual	Ham	24	4
	Phish	6	22

TMR-J48

		Classified	
		Ham	Phish
Actual	Ham	23	5
	Phish	12	16

The experimental results are presented below in two different figures. Performance is measured in terms of accuracy, which is the F_1 score. Accuracy was chosen as the performance measure due to how it incorporates both precision and recall.

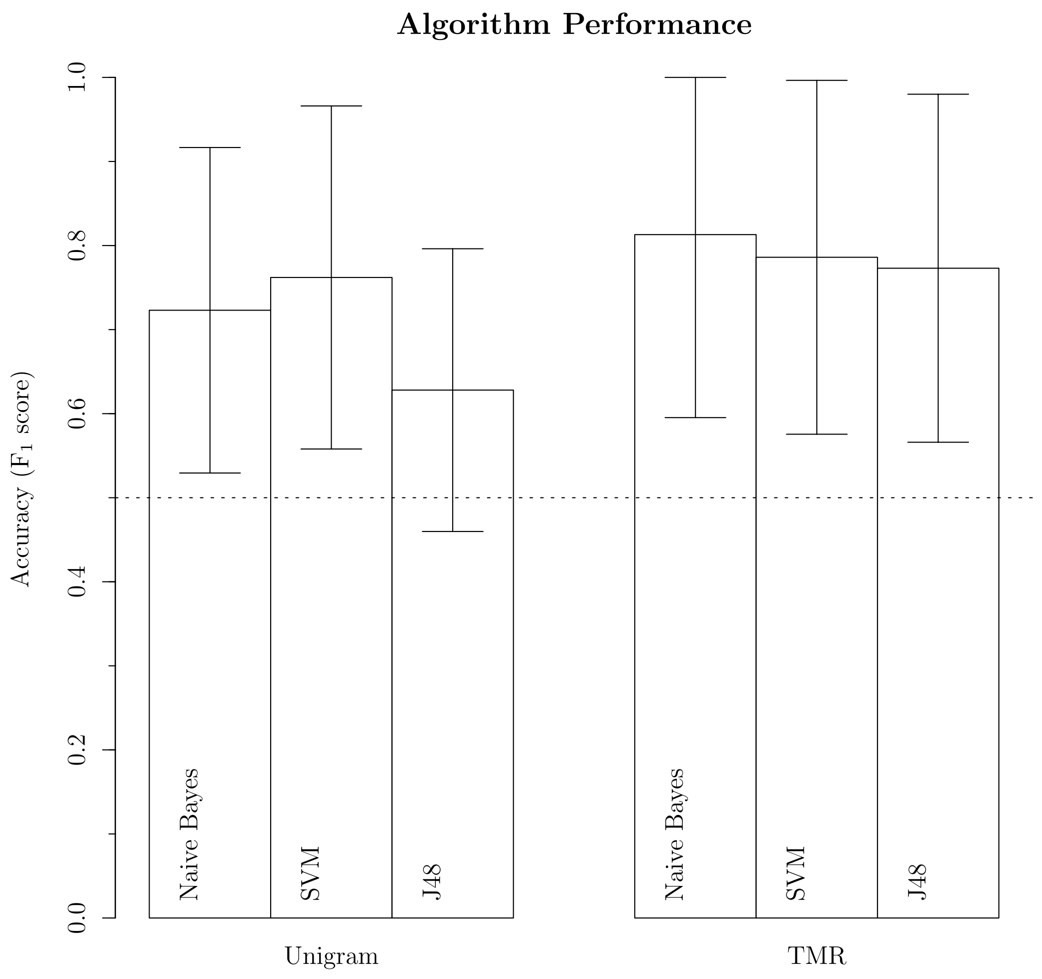


Figure 5.1. Accuracy of the machine learning models grouped by training data set.

Figure 5.1 shows performance of the algorithm experiments. The dashed horizontal line represents the accuracy of a random binary classifier (0.5 in this case). Intervals are for a 95% confidence level.

Clearly all the TMR-based models perform better than random. There is some statistical question about the performance of the unigram-based J48 model due to the lower end of the confidence interval dipping below 0.5. But that is not the primary concern of this research.

It also appears that the TMR-based models outperform the same unigram-based models. The mean accuracy of every TMR-based model is higher than any of the unigram-based models. Whether or not we can be confident in this conclusion is fodder for a statistical hypothesis test. Each of the three data set pairings shown in Figure 5.1 should be compared with the unigram model being the baseline and the TMR model being the experimental.

5.2 Statistical Hypothesis Test

The hypothesis, as expressed in (5.1), is that the TMR model ($\bar{x}_1$) performs only as well or worse than the unigram model ($\bar{x}_2$). The alternative hypothesis (5.1b) in this case is that the TMR model does in fact perform better than the unigram model (Moore, McCabe, & Craig, 2014).

$$H_0 : \bar{x}_1 \leq \bar{x}_2 \tag{5.1a}$$

$$H_a : \bar{x}_1 > \bar{x}_2 \tag{5.1b}$$

The statistical hypothesis test is the comparison of the two sample means. As an example of how the math works the values for the naive Bayes algorithm are used as input to the equations. Results for calculations of all three algorithms are given in Table 5.3.

The first step in the test is the calculation of the test statistic T as shown in (5.2). The end result is a test statistic value of $T = 0.236$.

$$\begin{aligned} T &= \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}}} \\ &= \frac{0.813 - 0.723}{\sqrt{\frac{0.102^2}{56} + \frac{0.126^2}{56}}} \\ &= \frac{0.09}{\sqrt{4.693 \cdot 10^{-4}}} \\ &= 4.155 \end{aligned} \tag{5.2}$$

Next, the degrees of freedom is calculated as shown in (5.3). The degrees of freedom value is necessary in later looking up the critical value. For the naive Bayes algorithm sample means the degrees of freedom ends up rounding to $v \approx 105$.

$$\begin{aligned} v &= \frac{(\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2})^2}{\frac{(\frac{s_1^2}{N_1})^2}{N_1-1} + \frac{(\frac{s_2^2}{N_2})^2}{N_2-1}} \\ &= \frac{(\frac{0.102^2}{56} + \frac{0.126^2}{56})^2}{\frac{(\frac{0.102^2}{56})^2}{55} + \frac{(\frac{0.126^2}{56})^2}{55}} \\ &= \frac{(4.693 \cdot 10^{-4})^2}{\frac{(1.858 \cdot 10^{-4})^2 + (2.835 \cdot 10^{-4})^2}{55}} \\ &= \frac{2.202 \cdot 10^{-7}}{2.089 \cdot 10^{-9}} \\ &= 105.429 \\ &\approx 105 \end{aligned} \tag{5.3}$$

Because the confidence intervals shown in Figures 5.1 and 5.2 are both calculated at the 95% level then so too will the statistical hypothesis tests here. For a 95% confidence interval the alpha level gets set to $\alpha = 0.05$. This alpha, along with the degrees of freedom calculated in (5.3) are used as parameters to look up the relevant t-score: $t_{1-\alpha,v} = t_{0.95,105} = 1.659$.

Recall back to the alternate hypothesis (5.1b) where $\bar{x}_1 > \bar{x}_2$. So in order to refute the null hypothesis (5.1a) $T > t_{1-alpha,v}$ must be true. In this example case of the naive Bayes data it is indeed correct that $T = 4.155 > t_{1-\alpha,v} = t_{0.95,105} = 1.659$. Therefore we can disprove the null hypothesis.

This same procedure is repeated for the subsequent SVM and J48 comparisons with all results compiled in Table 5.1.

Table 5.3

Calculations for comparing two sample means.

Algorithm	**T**	**Degrees of Freedom**	**Critical Value**
Naive Bayes	4.155	105	1.659
SVM	1.213	98	1.661
J48	5.982	95	1.661

In order to refute the null hypothesis(H_0), the test statistic T has to be greater than the critical value ($t_{1-\alpha,v}$). It is therefore trivial to see, using Table 5.3 as a reference, that in the cases of the naive Bayes classifier ($4.155 > 1.659$) and the J48 classifier ($5.982 > 1.661$) we reject the null hypothesis. However, in the case of the SVM classifier ($1.213 \not> 1.661$) we fail to reject the null hypothesis.

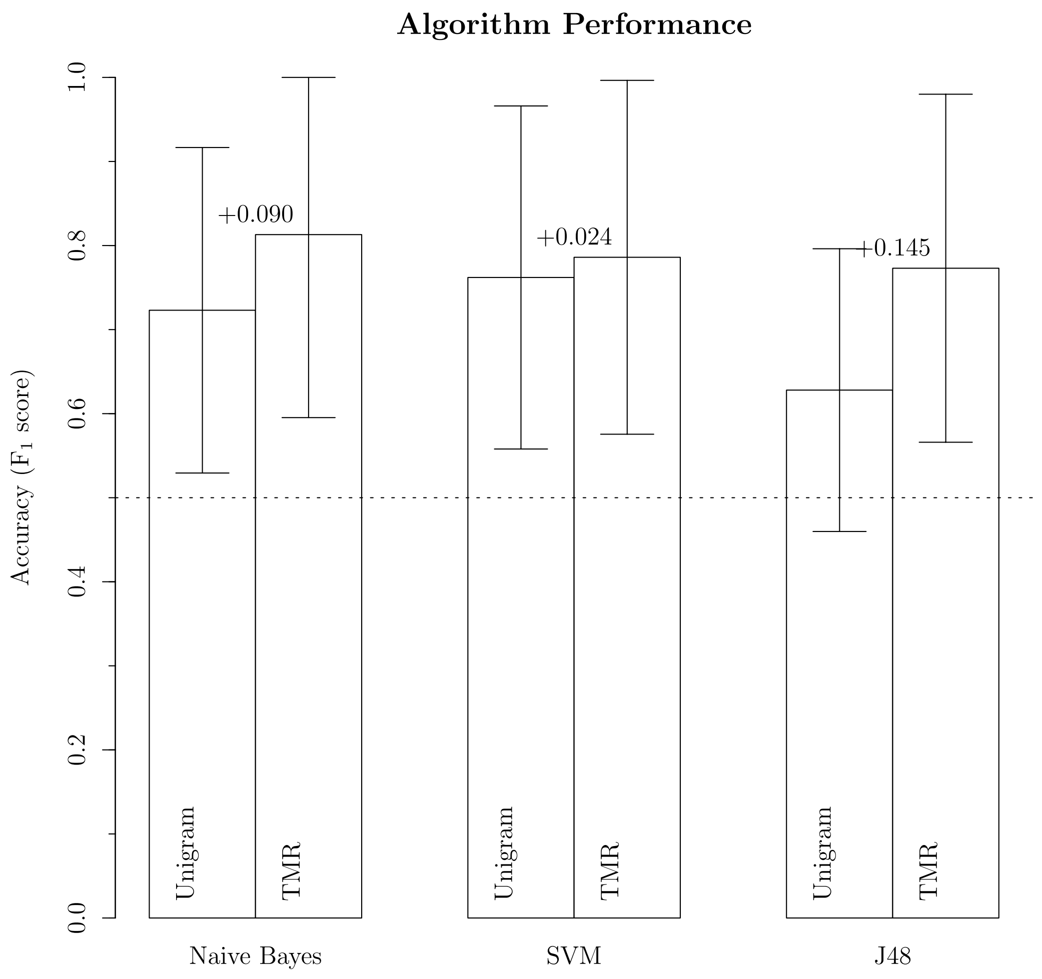


Figure 5.2. Accuracy of the machine learning models grouped by algorithm.

Figure 5.2 presents the same data seen in Figure 5.1 but with the accuracy scores grouped according to the algorithm their model was trained on. This second view makes it easier to observe the improved performance of the TMR-based models over their unigram-based counterparts. The numbers astride the middle points of the pairings quantify the difference in median values.

5.3 Misclassified Samples

Examining which files were misclassified by the machine learning algorithms provides insight into underlying problems.

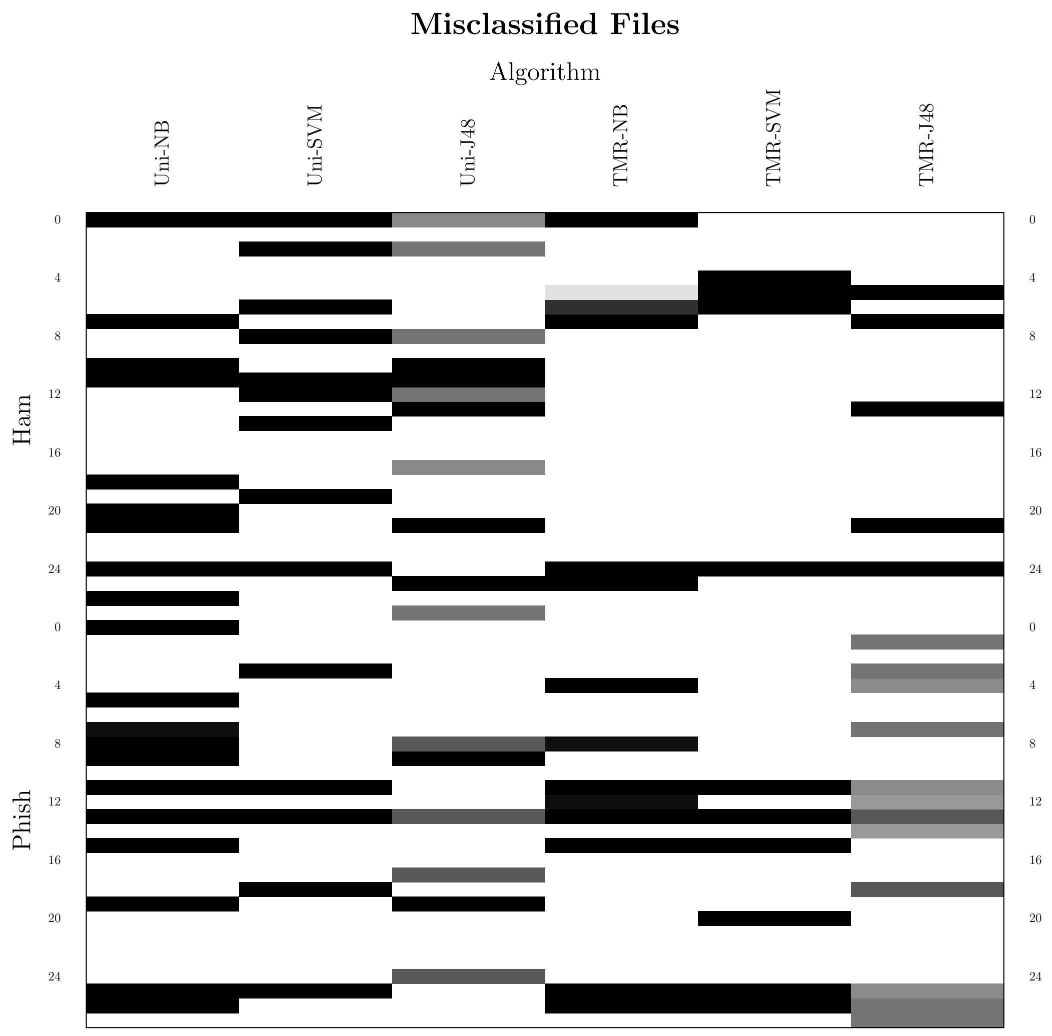


Figure 5.3. A plot of training files where the darkness of the cell corresponds to the confidence with which they were misclassified.

The graphic in Figure 5.3 shows at a glance how files were misclassified by each algorithm. A white rectangle indicates that a file was classified correctly while a colored rectangle indicates that the file was misclassified. The darkness of the rectangle gives an indication of the confidence with which the file was misclassified (Tufte, 1990, pp.91-93). It should be noted that the naive Bayes and J48 algorithms provide a probability with their classifications while the SVM algorithm only provides a boolean yes/no classification so all the probabilities are either 1.0 (misclassified) or 0.0 (correctly classified). A comprehensive table with all probabilities explicitly stated is included in Appendix A.

Several interesting features from Figure 5.3 jump out. First, the TMR-based algorithms misclassify ham emails far less than the unigram-based algorithms do. This is apparent by the lower density of colored rectangles in the TMR/Ham quadrant (upper-right) of the graphic. Second, the J48 classifiers are less confident when they misclassify files. The largest variety of shades of gray are seen in the unigram and TMR J48 classifiers. The naive Bayes classifier also return a probability with its classification but the unigram and TMR naive Bayes classifier cells are almost uniformly black.

Another, more quantifiable, way of analyzing the differences in misclassified files is by using the Jaccard index. The Jaccard index is a value between 0 and 1 that describes the similarity between two sets where 0 means completely dissimilar and 1 means identical (Jaccard, 1912):

$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$

Applying the Jaccard index to misclassified files means that the performance of two different algorithms can be compared to see if they're misclassifying the same or different files. Table 5.4 compares the same algorithm for its performance with the unigram data set versus the semantic data set. Jaccard index values tell us that the naive Bayes classifiers mostly misclassify the same files while the J48 decision tree classifier performs very different on the two data sets. The SVM classifier is evenly split with an index value of 0.500.

Table 5.4

Jaccard indices of misclassified files by algorithm.

Algorithm	**Intersection**	**Union**	**Jaccard**
Naive Bayes	9	14	0.643
SVM	5	10	0.500
J48	3	17	0.176

There was only one situation in which all six data set/algorithm combinations failed to correctly classify the email, and only two situations in which give of the six combinations failed. All three of these files are phishing email samples. Figure 5.2 and Figure 5.3 corroborate the fact that phishing samples are those most consistently misclassified across algorithms.

```
Please review the attached invoice and pay this invoice at your
earliest convenience.  Feel free to contact us if you have any
questions.

Thank you.
```

Figure 5.4. The email that was misclassified by all the models.

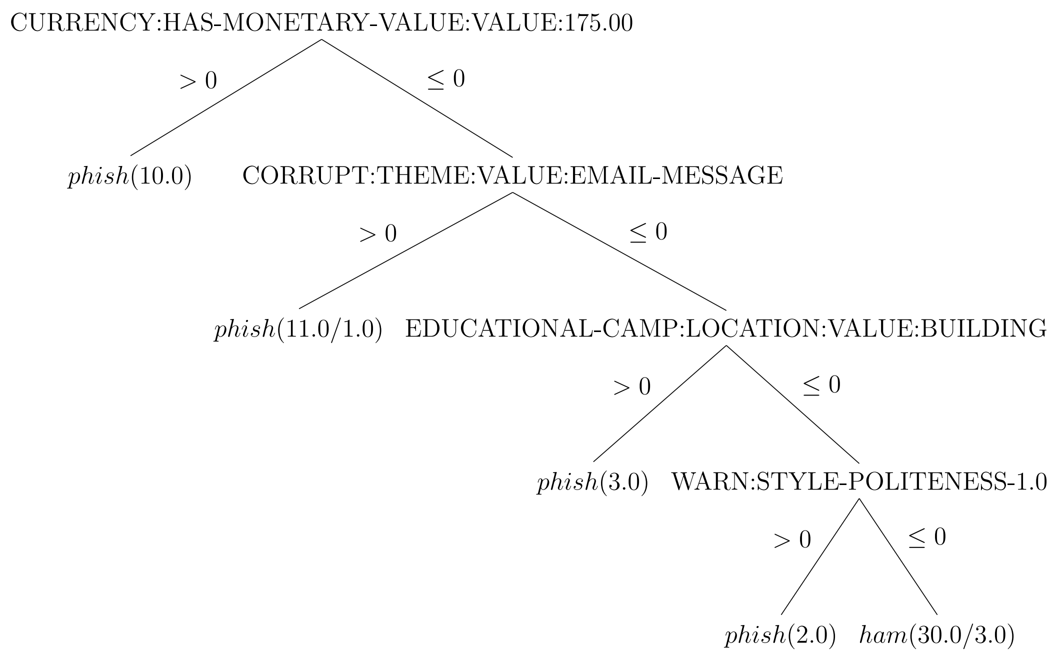

CURRENCY:HAS-MONETARY-VALUE:VALUE:175.00
> 0
≤ 0
phish(10.0)
CORRUPT:THEME:VALUE:EMAIL-MESSAGE
> 0
≤ 0
phish(11.0/1.0)
EDUCATIONAL-CAMP:LOCATION:VALUE:BUILDING
> 0
≤ 0
phish(3.0)
WARN:STYLE-POLITENESS-1.0
> 0
≤ 0
phish(2.0)
ham(30.0/3.0)

Figure 5.5. The J48 tree trained from semantic features.

The benefit of the output structure seen in Figure 5.5 is that it is human-readable. The same cannot be said about the J48 tree generated from unigrams in Figure 5.6. The following list describes the TMR tree nodes in descending order:

1. There is an amount of currency number 175 of some undefined denomination.

2. An email message was corrupted.

3. An organized, educational camp is being held inside of a particular building.

4. A polite warning.

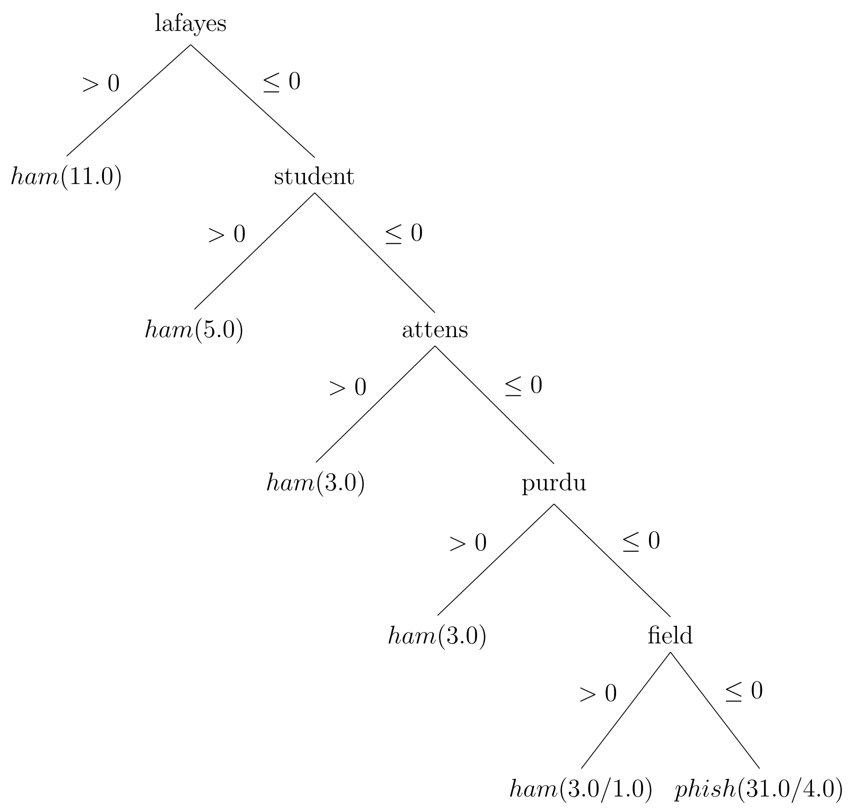


Figure 5.6. The J48 tree trained from uni-gram features.

The J48 tree model shown in Figure 5.6 differs significantly from the tree model shown in Figure 5.5 despite having the same number of nodes. Notice how the unigram strings are sometimes legible such as in the cases of "student" and "field" but other times the strings have garbled endings such as "lafayes", "attens", and "purdu." These garbled endings are artifacts of the stemming process applied prior to generating the machine learning model.

So unlike the TMR-trained tree of Figure 5.5 the unigram-trained tree doesn't provide individual nodes that are easily understood by a human. In this particular situation, however, the nodes when examined collectively can provide some information about the experiment. More specifically, all the unigram strings seem to be describing an academic theme centered around Purdue University. And the tree model uses this Purdue-centric terms to identify when an email is good (as opposed to identifying when it is phishing as in the TMR tree). This suggests that the known good emails overly represent language particular to Purdue and its activities. Further studies would do well to try a more diverse sampling of known good emails to avoid this problem.

5.4 Ontology

During the manual TMR generation process the accompanying ontology was generated from scratch. The result was 560 unique concepts. This ontology is parsimonious in the sense that it is the minimal ontology necessary to describe the language seen in the sample data. It is unlikely that the ontology as it currently exists is sufficient to describe new data without further additions. In this way the current ontology is not generalizable.

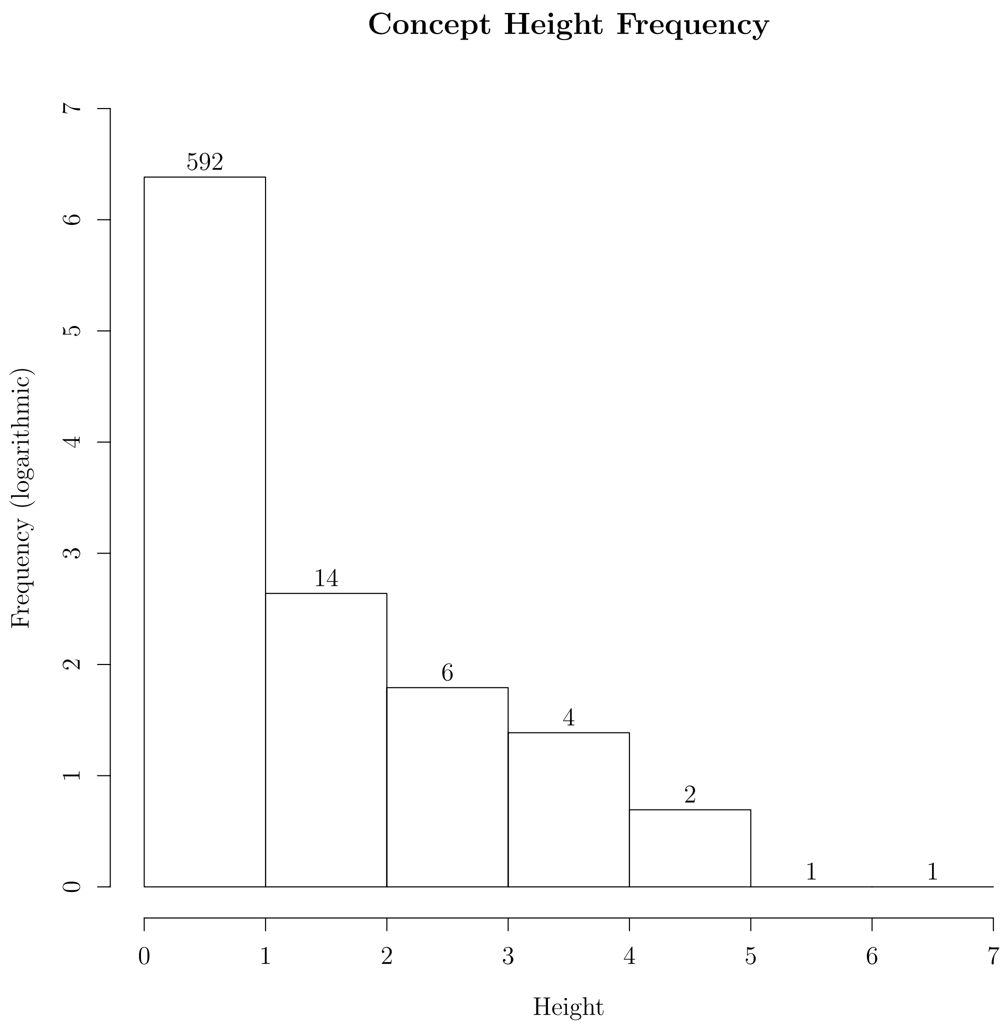


Figure 5.7. A histogram of the number of fact instances for a concept based on its height in the ontology.

Figure 5.7 shows how frequently (in terms of the logarithm) a concept is instantiated into a fact versus its height in the ontology. Height of a tree-like data structure begins at zero for the leaf nodes. Because the ontology created is single-parented (multiple inheritance is not utilized) then its taxonomic structure is a tree. The only concept with a height of seven would be the special purposed ALL concept, which should never be instantiated.

The histogram shoes an exponential curve with the leaf concepts occurring far more frequently than their more highly placed parents. This means that the TMRs generated by the research use the most fine-grained, specific concepts available instead of relying on more abstract concepts with a higher height. For example, if ANIMAL is a concept and DOG is also a concept that is the child of ANIMAL, and a leaf node concept, then this data would suggest that DOG would occur more frequently.

These data support the intuition that leaf node concepts should be seen more frequently than their parent concepts. An overuse of abstract concepts might suggest a problem in the TMR generation process that is causing an over-reliance on abstract concepts.

The following steps demonstrate the problems caused by using overly abstract concepts. It begins with Example 8. This is the input sentence to the OST system.

(8) "The cat climbs up the tree."

The OST system renders this sentence into the following frame:

```
(CLIMB-3
    (AGENT (VALUE (CAT-9)))
    (THEME (VALUE (TREE-102)))
)
```

But what would happen if instead of CAT and TREE, more abstract concepts were used? For this example it is assumed that CAT is the direct subclass of MAMMAL and TREE is the direct subclass of PLANT. The more abstract frame then becomes:

```
(CLIMB-3
    (AGENT (VALUE (MAMMAL-29)))
    (THEME (VALUE (PLANT-44)))
)
```

To better illustrate the problems created, translate the abstracted frame back into an English sentence. The result might be that seen in Example 9. This is a distinctly different sentence from Example 8 that served as input.

(9) "The mammal climbs up the plant."

The use of the frequency-versus-height statistic is one tool available to future acquisition efforts to help them analyze the health of their knowledge base.

5.5 Economics

Manually processing input texts into intermediate TMRs is a time consuming task. Part way through the processing stage of the research the idea was suggested to record the amount of time taken in processing each email. Processing times for the last twenty-three of the total fifty-six samples emails was recorded.

Figure 5.8 plots the size of each of the twenty-three emails against the time it took to process them. The straight line through the scatter plot is a linear regression to the data. The linear regression itself has a coefficient of determination (r^2) of 0.923, which suggests a good fit to the observations.

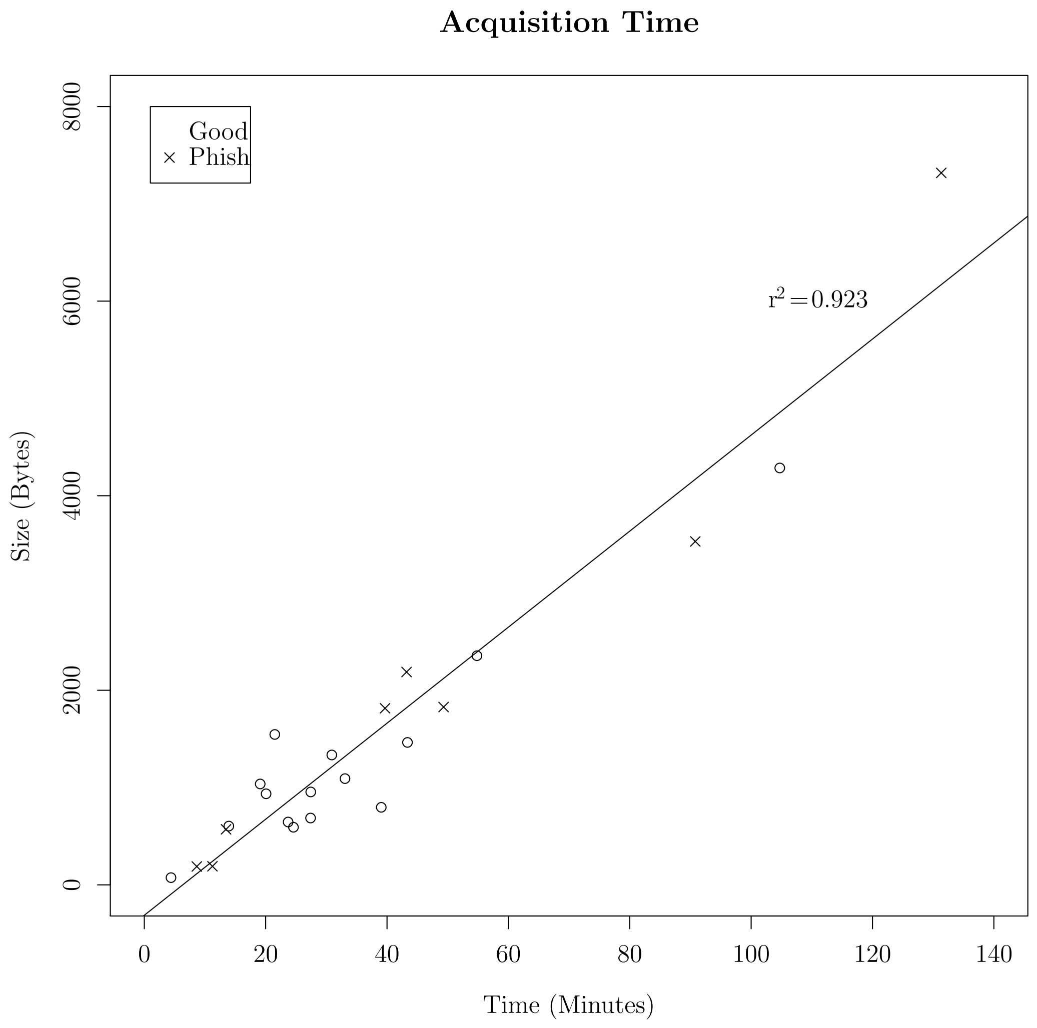


Figure 5.8. Amount of time taken to manually process sample emails.

The regression line from Figure 5.8 has potential use outside of this dissertation. Given a text corpus for manual processing, the regression can help predict how long it would take to process all the texts. This overall time can be divided by the number of available team members to determine how long such a project might take. Multiple the overall time by an hourly salary and that is the cost to process the corpus. Both values are useful items to include in a grant proposal.

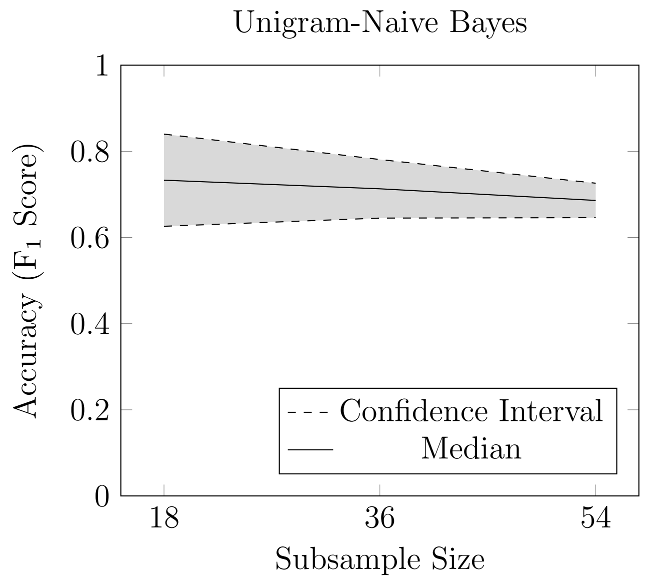


Figure 5.9. Unigram-Naive Bayes models converging as the sample size increases.

5.6 Convergence

The overall results use all fifty-six samples. There is a secondary question remains of how the different machine learning algorithms converge on their solutions as the number of samples increases.

Three sample sizes were chosen at regular intervals: 18, 36, and 54. All samples were evenly split between the two classes (9, 18, and 27 respectively). As in the earlier tests cross-validation is used as a means to avoid over-fitting but this time with K=3. The same six combinations of machine learning algorithms and datasets were trained and tested on these three subsamples.

5.7 Future Work

There are several different ways in which the research in this dissertation could be extended or improved. The obvious first choice is to get larger amount of

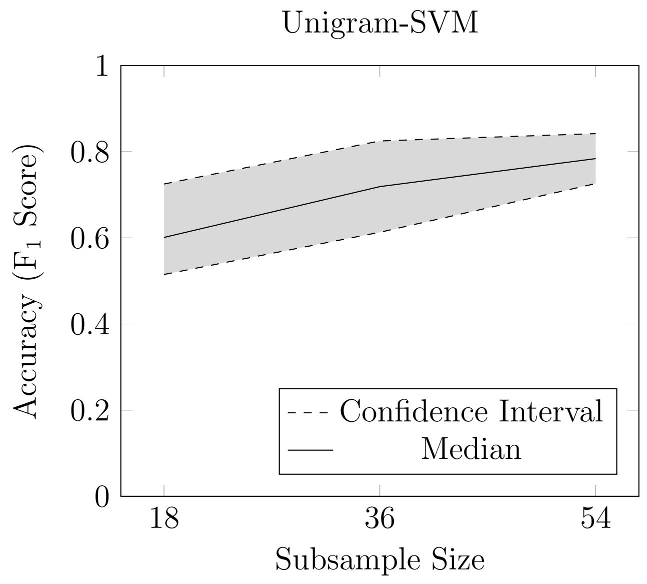


Figure 5.10. Unigram-SVM models converging as the sample size increases.

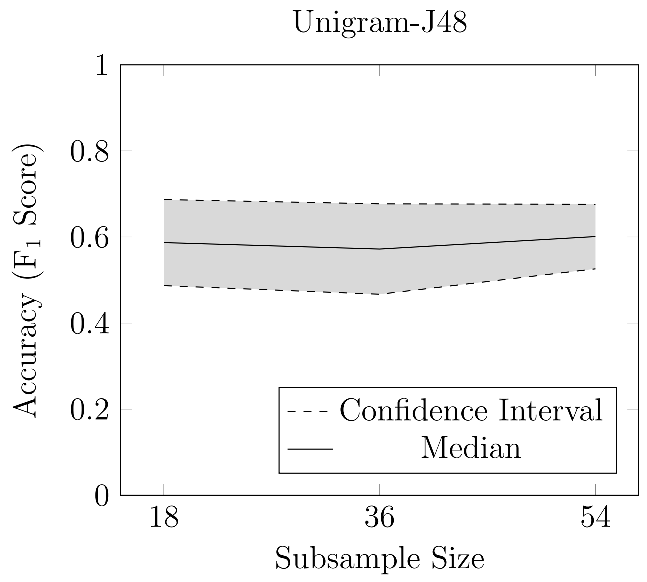


Figure 5.11. Unigram-J48 models converging as the sample size increases.

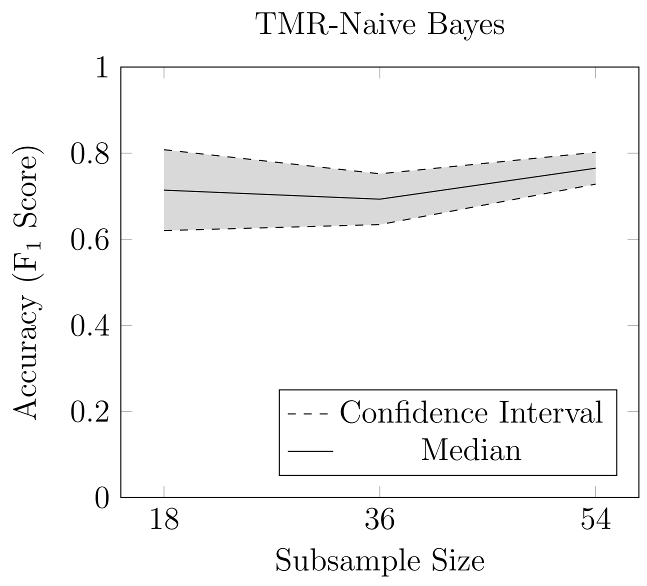


Figure 5.12. TMR-Naive Bayes models converging as the sample size increases.

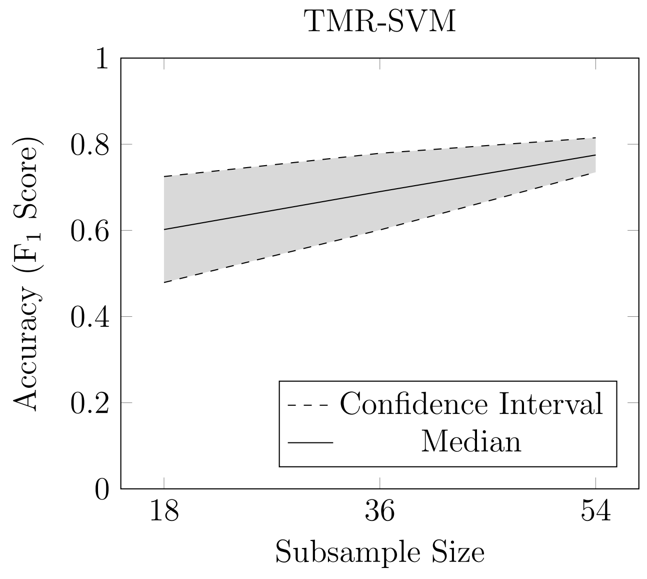


Figure 5.13. TMR-SVM models converging as the sample size increases.

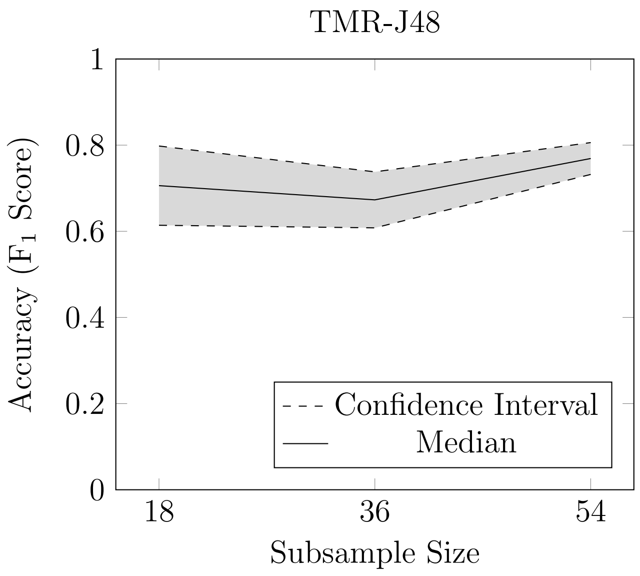


Figure 5.14. TMR-J48 models converging as the sample size increases.

data. Some of the choices made for this research provided unnecessary and possibly contributed to the small sample size.

5.7.1 Feature Space

This dissertation only scratches the surface of the features available for training ML models. The ML models tested here were only trained on one possible type of semantic feature.

1. Concept frequency
2. Local fact frequency
3. Two generations
4. Lexical mappings

Concept frequency Counting the frequency of individually occurring concepts would be the simplest feature to take from a semantic understanding of a phishing email. Concept counts weren't utilized as features in this research due to the possibility of them being virtually identical to unigrams. Recall that the text used to generate the unigram language model was stripped of stop words. All the remaining nouns, and the main verbs, would each have one concept that they map to. Ignoring the auxiliary verbs, adjectives, and adverbs, this would have provided a large degree of overlap between the unigram and TMR models.

Local fact frequency The currently used approach to identifying fact instantiated from concepts is to give them globally unique identifiers. Thus, HUMAN-43 in one document must be the same individual as HUMAN-43 in another document. But there is also the possibility of facts with identifiers that are unique only within the context of a specific text. The formalism suggested here to to append an ascending letter sequence instead of a number to the end of a fact. So a text referring to three different people would have HUMAN-A, HUMAN-B, and

HUMAN-C in this text-local naming scheme as opposed to HUMAN-111, HUMAN-112, and HUMAN-113 in the globally unique naming scheme.

This can help in identifying relations between individuals in an event. For instance, the code block below shows a SPEAK event with globally unique facts.

```
(SPEAK-7
        (AGENT (VALUE (HUMAN-88)))
        (BENEFICIARY (VALUE (HUMAN-97)))
)
```

Compare that to this following code block, which uses only the concepts used to instantiate the facts. If this frame is decomposed in quadruples then there is ⟨SPEAK, AGENT, VALUE, HUMAN⟩ and ⟨SPEAK, BENEFICIARY, VALUE, HUMAN⟩. There is missing information to inform the machine learning software that the two SPEAK instances are in fact the same one, and that the two HUMAN instances are necessarily two different individuals.

```
(SPEAK
        (AGENT (VALUE (HUMAN)))
        (BENEFICIARY (VALUE (HUMAN)))
)
```

Text-specific fact labeling might solve that problem. Code block 5.7.1 shows what this solution would look like.

```
(SPEAK-A
        (AGENT (VALUE (HUMAN-A)))
        (BENEFICIARY (VALUE (HUMAN-B)))
)
```

The resulting decomposed quadruples are then ⟨SPEAK-A, AGENT, VALUE, HUMAN-A⟩ and ⟨SPEAK-A, BENEFICIARY, VALUE, HUMAN-B⟩, which describe some of the distinctions between the facts.

All of this begs the question of how to label these text-specific facts. The initially proposed scheme is to label the facts with ascending letters according to their descending frequency. So the fact appended with "-A" is the most frequently occurring fact of that particular concept, "-B" is the second most frequently occurring fact, so on and so forth.

Two generations The approach of decomposing intermediate TMRs into quadruples used in this research could be described as looking at a single generation of the TMR. A single generation in this case is connecting from a root concept to the first filler. But a two generation feature would decompose an intermediate TMR such that each feature connects from a root concept to the first filler and then on to a second filler, that is in turn a filler of the first filler in the given feature. This would be a 7-tuple, or a septuple.

```
(BUY−7
    (AGENT  (VALUE  (HUMAN−3
        (PLAYS−ROLE  (VALUE  (STUDENT− 19)))
    )))
)
```

The event in the above code block in the one generation ML feature scheme would give two quadruples: ⟨BUY-7, AGENT, VALUE, HUMAN-3⟩ and ⟨HUMAN-3, PLAYS-ROLE, VALUE, STUDENT-19⟩. But in the two generation scheme the same event generates only a single septuple: ⟨BUY-7, AGENT, VALUE, HUMAN-3, PLAYS-ROLE, VALUE, STUDENT-19⟩. This two generation scheme captures what might be necessary information that the one generation scheme does not; that it is in fact a student making the purchase and not just a generic person. This improved scheme is important for capturing formalisms such as social roles where concepts used as fillers are further specified.

This multi-generational ML feature begs the question of when are there too many generations. Increasing the number of generations used to produce ML features could cause an explosion in the overall number of features. This is a situation of diminishing returns. The question of where to stop the number of generations is to be determined experimentally. This is also a place where ML might offer assistance with techniques like information gain and principle component analysis (PCA) identifying which ML features are contributing the most to a classifier's performance.

Lexical mappings are the mappings from the word as it appears in the text and the fact it instantiated in the TMR. Lexical mappings were not recorded for this research. It is known in NLP research that for polysemous words there is usually only one sense of that word used in any given document **??**. A new research question might be: Do phishing emails use one particular sense of a word more often than non-phishing emails, and might that be an effective way to distinguish the two?

5.7.2 Graph Theory

A proper, finished TMR is essentially a graph. It stands to reason then that tools used to analyze graphs, such as social network analysis, might be useful in answering questions about the knowledge represented in a TMR.

Future work should investigate how the network structures between good and phishing emails differ. Tools like Gephi and R facilitate social network analysis. But to conduct a proper examination would require experience with both graph theory and the generation of ML features from graph structures, which were both lacking at the time of this research.

5.8 Summary

Overall this research is considered a success. It met its primary goal of building ML classifiers on semantic features in such a way as the results were not random. Furthermore, the majority of these classifiers performed better than their unigram counterparts on the same data.

Not all avenues that were explored proved to be fruitful. The data collected to conduct the research was delimited to be emails with PDF attachments. The question was if the language in the emails that referred to the attachments might be key in distinguishing good from phishing. No such connection was identified. There is possible future research to be done in comparing phishing emails with PDF malware attachments to other phishing emails with PDF attachments that are not malware. The phishing email corpus collected for this particular research project totalled just twenty-eight with only four of these phishing emails having malicious attachments. Larger amounts of data must be examined to find sufficient samples for such a future work.

This research intentionally focused on just the language used in the bodies of email messages. No real, commercial product would restrict itself to such a constrained set of indicators. This delimitation may have artificially lowered the ML scores seen in this chapter. But the goal of this research was never to create a contrived situation for the purpose of incrementally increasing a simple score.

This research also serves the purpose of showing that OST has practical applications. Phishing is just one particular application. Stylometry, anonymization, and sentiment analysis are all problems that might benefit from a semantic approach.

OST is a theory undergoing continued development and refinement. While machine learning might not be able to help develop that theory it could serve as a tool for measuring progress. As new ideas as formalized, the expanded OST resource could undergo a kind of regression testing to compare ML results from the new structures to the ML results of the old structures.

Perhaps it could be said that successful research just leads to more questions?

APPENDICES

	File	Uni-NB	Uni-SVM	Uni-J48	TMR-NB	TMR-SVM	TMR-J48
Ham	00	1.000	1.000	0.875	1.000		
	01						
	02		1.000	0.913			
	03						
	04					1.000	
	05			0.622	1.000	1.000	
	06		1.000		0.987	1.000	
	07	1.000			1.000		1.000
	08		1.000	0.913			
	09						
	10	1.000		1.000			
	11	1.000	1.000	1.000			
	12		1.000	0.913			
	13			1.000			1.000
	14		1.000				
	15				1.000		
	16						
	17			0.875			
	18	1.000					
	19		1.000				
	20	1.000					

	21	1.000		1.000			1.000
	22						
	23						
	24	1.000	1.000		1.000	1.000	1.000
	25			1.000	1.000		
	26	1.000					
	27			0.913			
	00	1.000					
	01						0.913
	02						
	03		1.000				0.913
	04				1.000		0.870
	05	1.000					
	06						
	07	0.999					0.913
	08	1.000		0.952	0.999		
	09	1.000		1.000			
	10						
	11	1.000	1.000		1.000	1.000	0.870
	12				0.999		0.840
Phish	13	1.000	1.000	0.952	1.000	1.000	0.952
	14						0.840
	15	1.000			1.000	1.000	
	16						
	17			0.952			
	18		1.000				0.952
	19	1.000		1.000			
	20					1.000	
	21						

22						
23						
24			0.952			
25	1.000	1.000		1.000	1.000	0.870
26	1.000			1.000	1.000	0.913
27						0.913

CHAPTER B. ANTLR GRAMMAR FOR INTERMEDIATE TMRS

This is the ANTLR grammar file that defines how an intermediate TMR file used in this research must look. ANTLR is a parser generation tool for the Java programming language. ANTLR will take this file as input and automatically generate the necessary Java code for a parser. A programmer can then extend the relevant classes in the ANTLR API to construct a walker that traverse the abstract syntax tree created by the ANTLR-generated parser.

The grammar in this file is compatible with ANTLR version 4 (Parr, 2013). ANTLR grammar definition files are not backwards compatible with previous versions of ANTLR.

```
grammar Tmr;


WS : [ \t\r\n]+ -> skip;
LPAREN : '(';
RPAREN : ')';
NUMBER : [0-9]+('.' [0-9]+)?;
RANGE_OPERATOR : '>' | '>=' | '<=' | '<';
RANGE : RANGE_OPERATOR NUMBER;
BOOLEAN : 'TRUE' | 'FALSE';
TIME : 'TIME';
TIME_START : 'START';
TIME_END : 'END';
FACET_TYPE : 'VALUE' | 'DEFAULT-MEASURE' | 'NOT';
SET : 'SET-' NUMBER;
CARDINALITY : 'CARDINALITY';
```

```
LOGIC : 'LOGIC';
SUBSET : 'SUBSET-OF';
LOGIC_TYPE : 'AND' | 'OR' | 'NOT' | 'XOR';
MODALITY : 'MODALITY';
MODALITY_TYPE : 'EPISTEMIC' | 'DEONTIC' | 'VOLITIVE'
        | 'POTENTIAL' | 'EPITEUCTIC' | 'EVALUATIVE'
        | 'SALIENCY';
STYLE : 'STYLE';
STYLE_TYPE : 'POLITENESS' | 'COLOR' | 'FORMALITY';
ASPECT : 'ASPECT';
PHASE : 'PHASE';
ITERATION : 'ITERATION';
STRING : '"' ~["]* '"';
LABEL : [A-Z]+ ('-' [A-Z]+)*;
FACT : LABEL '-' NUMBER;

tmr : statement+;

statement : LPAREN statement_core RPAREN;

statement_core : (FACT | set)
        (slot | time | modality | style | aspect)*;

slot : LPAREN LABEL facet+ RPAREN;

facet : LPAREN FACET_TYPE filler+ RPAREN;

filler : LPAREN filler_value+ RPAREN;
```

```
filler_value : LPAREN filler_value RPAREN
    | statement_core | NUMBER | RANGE | LABEL | STRING
    ;

set : SET set_parameter*;

set_parameter : cardinality | completeness | type | includes
        | excludes | logic | subset;

cardinality : LPAREN CARDINALITY LPAREN (RANGE+ | NUMBER)
        RPAREN RPAREN;

completeness : LPAREN 'COMPLETE' LPAREN BOOLEAN RPAREN RPAREN;

type : LPAREN 'TYPE' LPAREN LABEL RPAREN RPAREN;

includes : LPAREN 'INCLUDES' LPAREN (FACT | LABEL | SET)+
        RPAREN RPAREN;

excludes : LPAREN 'EXCLUDES' LPAREN (FACT | LABEL | SET)+
        RPAREN RPAREN;

logic : LPAREN LOGIC LPAREN LOGIC_TYPE RPAREN RPAREN;

subset : LPAREN SUBSET LPAREN SET RPAREN RPAREN;

time : LPAREN TIME (absolute_time | time_range | relative_time
        | time_duration) RPAREN;
```

```
absolute_time : LPAREN STRING RPAREN;

time_range : LPAREN start_time RPAREN (LPAREN end_time RPAREN)?
    | LPAREN end_time RPAREN;

start_time : TIME_START LPAREN
        (STRING | reference_time | relative_time) RPAREN;

end_time : TIME_END LPAREN (STRING | reference_time) RPAREN;

relative_time : LPAREN ('PAST' | 'PRESENT' | 'FUTURE') RPAREN;

reference_time : (RANGE_OPERATOR? FACT '.' TIME
        ('.' (TIME_START | TIME_END))?)+;

time_duration : LPAREN 'DURATION' LPAREN LABEL LPAREN NUMBER
        RPAREN RPAREN RPAREN;

modality_pair: LPAREN MODALITY_TYPE LPAREN (NUMBER | RANGE)
        RPAREN RPAREN;

modality : LPAREN MODALITY (modality_pair)+ RPAREN;

style : LPAREN STYLE (LPAREN STYLE_TYPE LPAREN (NUMBER | RANGE)
        RPAREN RPAREN)+ RPAREN;

aspect : LPAREN ASPECT (iteration | phase) RPAREN;

iteration : LPAREN ITERATION LPAREN (NUMBER | 'FIRST' | 'LAST')
```

```
          RPAREN RPAREN;

phase  :  LPAREN PHASE LPAREN
           ('BEGIN'  |  'CONTINUE'  |  'END'  |  'OTHER')  RPAREN RPAREN;
```

LIST OF REFERENCES

LIST OF REFERENCES

Abraham, D., & Raj, N. S. (2014). Approximate string matching algorithm for phishing detection. In *International conference on advances in computing, communications and informatics (ICACCI), 2014* (pp. 2285–2290).

Adobe Product Security Incident Response Team. (2012, April 10). *Background on security bulletin APSB12-08* [Blog Post]. Online. Retrieved 2016-06-14, from `http://blogs.adobe.com/security/2012/04/background-on-security-bulletin-apsb12-08.html`

Afroz, S., & Greenstadt, R. (2011). Phishzoo: Detecting phishing websites by looking at them. In *2011 fifth ieee international conference on semantic computing (icsc)* (pp. 368–375).

Aggarwal, A., Kumar, V., & Sundarsan, S. (2014). Identification and detection of phishing emails using natural language processing techniques. *Proceedings of the 7th International Conference on Security of Information and Networks*.

Aggarwal, A., Rajadesingan, A., & Kumaraguru, P. (2012). PhishAri: Automatic realtime phishing detection on Twitter. In *ecrime researchers summit (ecrime), 2012* (pp. 1–12).

Aleph One. (1996, August 11). Smashing the stack for fun and profit. *Phrack*, *7*(49). Retrieved from `http://insecure.org/stf/smashstack.html`

Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., & Spyropoulos, C. D. (2000). An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages. In *Proceedings of the 23rd annual international acm sigir conference on research and development in information retrieval* (pp. 160–167).

Anti-Phishing Working Group. (2015). *APWG archive* [Corpus]. Retrieved from `https://archive.apwg.org/index.php`

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. *Scientific American*, *284*(5), 28–37.

Bishop, M. (2002). *Computer security: art and science*. Boston: Addison-Wesley.

Brachman, R. J. (1983). What IS-A is and isn't: An analysis of taxonomic links in semantic networks. *Computer*, *16*(10), 30–36.

Breiman, L. (1998). Arcing classifier (with discussion and a rejoinder by the author). *The Annals of Statistics*, *26*(3), 801–849.

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5–32.

Chang, C., & Lin, C. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, *2*, 27:1–27:27. (Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`)

Chen, J., & Guo, C. (2006). Online detection and prevention of phishing attacks. In *First international conference on communications and networking in china, 2006* (pp. 1–7).

Chen, J. C., & Li, B. (2015). *Evolution of exploit kits: Exploring past trends and current improvements* (Research Paper). Irving, Texas: Trend Micro. Online. Retrieved 2016-06-17, from `https://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp-evolution-of-exploit-kits.pdf`

Chou, N., Ledesma, R., Teraguchi, Y., Boneh, D., & Mitchell, J. C. (2004, February). Client-side defense against web-based identity theft. In *11th annual network and distributed system security symposium.* San Diego.

Cohen, W. W. (2015). *Enron email dataset.* Corpus. Retrieved 2015-05-08, from `https://www.cs.cmu.edu/~./enron/`

Cook, D. L., Gurbani, V. K., & Daniluk, M. (2009, January). Phishwish: a simple and stateless phishing filter. *Security and Communication Networks*, *2*(1), 29–43.

Cormack, G. V., & Lynam, T. R. (2005). *TREC 2005 spam public corpora* [Corpus]. Retrieved from `http://plg.uwaterloo.ca/cgi-bin/cgiwrap/gvcormac/foo`

Cortes, C., & Vapnik, V. (1995, September). Support-vector networks. *Machine Learning*, *20*(3), 273–297.

Crocker, D. H. (1982, August 13). *Standard for ARPA internet text messages* (Online). Fremont: Internet Engineering Task Force. Retrieved from `https://www.ietf.org/rfc/rfc0822.txt`

Dullian, T. (2011). *Exploitation and state machines: Programming the "weird machine", revisited* [Presentation]. Online. Miami Beach. Retrieved from `http://titanium.immunityinc.com/infiltrate/archives/Fundamentals%5fof%5fexploitation%5frevisited.pdf`

Dunlop, M., Groat, S., & Shelly, D. (2010). GoldPhish: Using images for content-based phishing analysis. In *The fifth international conference on internet monitoring and protection* (p. 123-128).

Dykstra, J. (2016). *Essential cybersecurity science: Build, test, and evaluate secure systems.* Sebastopol, CA: O'Reilly Media.

Eryq, & Skoll, D. (2015). *MIME::Parser* [Software]. Retrieved 2015-04-22, from `http://search.cpan.org/dist/MIME-tools/lib/MIME/Parser.pm`

Falk, C., & Stuart, L. (2016). Meaning-based machine learning for information assurance. In *Proceedings of the twenty-ninth international Florida artificial intelligence research society conference* (pp. 264–269). AAAI Press.

Fellbaum, C. (2002). On the semantics of troponymy. In R. Green, C. A. Bean, & S. H. Myaeng (Eds.), *The semantics of relationships: An interdisciplinary perspective* (pp. 23–34). Kluwer Academic Publishers.

Fette, I., Sadeh, N., & Tomasic, A. (2007). Learning to detect phishing emails. In *Proceedings of the 16th international conference on world wide web* (pp. 649–656).

Guarino, N., & Welty, C. (2002a). Evaluating ontological decisions with ontoclean. *Communications of the ACM*, *45*(2), 61–65.

Guarino, N., & Welty, C. (2002b). The semantics of relationships: An interdisciplinary perspective. In R. Green, C. A. Bean, & S. H. Myaeng (Eds.), (pp. 111–126). Dordrecht: Kluwer Academic Publishers.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software: An update. *SIGKDD Explorations*, *11*(1).

Howard, F. (2012). *Exploring the Blackhole exploit kit* (Technical Paper). Abingdon: SophosLabs, UK. Retrieved from `https://sophosnews.files.wordpress.com/2012/03/blackhole%5fpaper%5fmar2012.pdf`

Hutchins, E. M., Cloppert, M. J., & Amin, R. M. (2011, March). Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Proceedings of the 6th International Conference on i-Warfare and Security*, *1*.

Jaccard, P. (1912). The distribution of the flora in the alpine zone. *New Phytologist*, *11*(2), 37–50.

James, L. (2005). *Phishing exposed.* Rockland: Syngress Publishing.

Java, A., Nirenburg, S., McShane, M., Finin, T., English, J., & Joshi, A. (2007). Using a natural language understanding system to generate semantic web content. *International Journal on Semantic Web and Information Systems*, *3*(4), 50–74.

John, G. H., & Langley, P. (1995). Estimating continuous distributions in Bayesian classifiers. In *Eleventh conference on uncertainty in artificial intelligence* (pp. 338–345). San Mateo: Morgan Kaufmann.

Jurafsky, D., & Martin, J. H. (2009). *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition* (Second ed.). Upper Saddle River, NJ: Pearson Prentice Hall.

Kirda, E., & Kruegel, C. (2005). Protecting users against phishing attacks with antiphish. In *29th annual international computer software and applications conference, 2005* (pp. 517–524).

Kojm, T. (2013). *Clam AntiVirus version 0.98 user manual* [Manual]. Online. San Jose. Retrieved from `https://github.com/vrtadmin/clamav-faq/raw/master/manual/clamdoc.pdf`

Koziol, J., Litchfield, D., Aitel, D., Anley, C., Eren, S., Mehta, N., & Hassell, R. (2004). *The shellcoder's handbook.* Indianapolis: Wiley.

Krebs, B. (2014). *Spam nation: the inside story of organized cybercrime-from global epidemic to your front door.* Naperville: Sourcebooks.

Krishnamurthy, R., & Orasan, C. (2002). *A linguistic investigation of the junk emails.* Corpus. Retrieved from `http://rgcl.wlv.ac.uk/resources/junk-emails/corpus-no-duplications.tar.gz`

Kumaraguru, P., Sheng, S., Acquisti, A., Cranor, L. F., & Hong, J. (2010). Teaching Johnny not to fall for phish. *ACM Transactions on Internet Technology*, *10*(2), 1–31.

Lakoff, G. (1987). *Women, fire, and dangerous things.* Chicago: University of Chicago Press.

Langberg, M. (1995, September 8). AOL acts to thwart hackers. *Mercury News*, 1C. Retrieved 2016-06-6, from `http://simson.net/clips/1995/95.SJMN.AOL%5fHackers.html`

L'Huillier, G., Hevia, A., Weber, R., & Ros, S. (2010). Latent semantic analysis and keyword extraction for phishing classification. In *2010 IEEE international conference on intelligence and security informatics.*

Lovins, J. B. (1968). Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, *11*(1), 23–31.

Masolo, C., Vieu, L., Bottazzi, E., Catenacci, C., Ferrario, R., Gangemi, A., & Guarino, N. (2004). Social roles and their descriptions. In *KR* (pp. 267–277). AAAI Press.

Mayer, D. (2012, November 11). *ratio of bugs per line of code* [Blog Post]. Online. Retrieved 2016-06-17, from `http://www.mayerdan.com/ruby/2012/11/11/bugs-per-line-of-code-ratio/`

McDonough, C. (2000). *Complex events in an ontological-semantic natural language processing system.* Unpublished doctoral dissertation, Purdue University, West Lafayette.

McDonough, C. J. (2000). *Complex events in an ontological-semantic natural language processing system.* Unpublished doctoral dissertation, Purdue University, West Lafayette.

McShane, M., & Babkin, P. (2015). Automatic ellipsis resolution: Recovering covert information from text. In *Proceedings of the twenty-ninth AAAI conference on artificial intelligence* (pp. 572–578). AAAI Press.

Microsoft Malware Protection Center. (2016). *Ransomware facts* [Web Page]. Online. Retrieved 2016-06-14, from `https://www.microsoft.com/en-us/security/portal/mmpc/shared/ransomware.aspx`

Microsoft Security TechCenter. (2012, May). *Security bulletin severity rating system.* Online. Retrieved 2016-06-17, from `https://technet.microsoft.com/en-US/security/gg309177.aspx`

Mitnick, K. D., & Simon, W. L. (2002). *The art of deception: Controlling the human element of security.* Indianapolis: Wiley Publishing.

MITRE. (2016). *Common vulnerabilities and exposures* [Search Result]. Online. Retrieved 2016-06-14, from `https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=PDF`

Moore, D. S., McCabe, G. P., & Craig, B. A. (2014). *Introduction to the practice of statistics* (Eighth ed.). New York: W.H. Freeman and Co.

Nirenburg, S., & Raskin, V. (2004). *Ontological semantics.* Cambridge, MA: The MIT Press.

Oard, D., Webber, W., Kirsch, D., & Golitsynskiy, S. (2015, February 16). *Avocado research email collection.* Corpus. Retrieved from `https://catalog.ldc.upenn.edu/LDC2015T03`

O'Harrow, R., Jr. (2016, March 27). How Clinton's email scandal took root. *Washington Post*. Retrieved from `https://www.washingtonpost.com/investigations/how-clintons-email-scandal-took-root/2016/03/27/ee301168-e162-11e5-846c-10191d1fc4ec%5fstory.html`

Onyshkevich, B., & Nirenburg, S. (1995). A lexicon for knowledge-based MT. *Machine Translation*, *10*, 5–57.

Open Web Application Security Project. (2013, December 31). *Code injection* [Wiki Page]. Online. Retrieved 2016-06-17, from `https://www.owasp.org/index.php/Code%5fInjection`

Park, G. (2013). *Text-based phishing detection using a simulation model.* Unpublished master's thesis, Purdue University, West Lafayette.

Park, G., & Taylor, J. M. (2015). *Using syntactic features for phishing detection* [Preprint]. Online. Retrieved 2016-04-28, from `http://arxiv.org/abs/1506.00037`

Park, G., Taylor, J. M., Stuart, L. M., & Raskin, V. (2014). Comparing machine and human ability to detect phishing emails. In *2014 ieee international conference on systems, man, and cybernetics* (pp. 2322–2327).

Parr, T. (2013). *The definitive ANTLR 4 reference* (2nd ed.). Pragmatic Bookshelf.

PhishMe. (2016, June 1). *Q1 2016 malware review* [White Paper]. Online. Retrieved 2016-06-02, from `http://phishme.com/phishme-q1-2016-malware-review/`

Prakash, P., Kumar, M., Kompella, R., & Gupta, M. (2010, March). PhishNet: Predictive blacklisting to detect phishing attacks. In *INFOCOM, 2010 proceedings IEEE* (pp. 1–5).

Pribbenow, S. (2002). Meronymic relationships: From classical mereology to complex part-whole relations. In R. Green, C. A. Bean, & S. H. Myaeng (Eds.), *The semantics of relationships: An interdisciplinary perspective* (Vol. 3, pp. 35–50). Kluwer Academic Publishers.

Princeton University. (2010). *About WordNet* [Web Page]. Online. Princeton. Retrieved from `http://wordnet.princeton.edu`

Quinlan, J. R. (1993). *C4.5: Programs for machine learning.* San Francisco: Morgan Kaufmann Publishers.

Radford, A. (2009). *Analysing English sentences: A minimalist approach.* Cambridge: Cambridge University Press.

Ramathan, V., & Wechsler, H. (2012). phishGILLNET–phishing detection methodology using probabilistic latent semantic analysis, AdaBoost, and co-training. *EURASIP Journal of Information Security*, *2012*(1).

Ramesh, G., & Krishnamurthi, I. (2014, September). PhishTackle–a web services architecture for anti-phishing. *Cluster Computing*, *17*(3), 1051–1068.

Rapid7. (2012, December 19). *Social engineering security and phishing with Metasploit.* Retrieved from `http://www.rapid7.com/resources/videos/phishing-with-metasploit.jsp`

Raskin, V., Hempelmann, C. F., Triezenberg, K. E., & Nirenburg, S. (2001). Ontology in information security: a useful theoretical foundation and methodological tool. In *Proceedings of the 2001 workshop on new security paradigms* (pp. 53–59). ACM.

Raskin, V., Nirenburg, S., Atallah, M. J., Hempelmann, C. F., & Triezenberg, K. E. (2002). Why NLP should move into IAS. In *Proceedings of the 2002 COLING workshop: A roadmap for computational linguistics-volume 13* (pp. 1–7). Association for Computational Linguistics.

Raskin, V., & Taylor, J. (2013). A fresh look at semantic natural language information assurance and security: NL IAS from watermarking and downgrading to discovering unintended inferences and situational conceptual defaults. In B. Akhgar & H. R. Arabnia (Eds.), *Emerging trends in ICT security.* Elsevier.

Recanati, F. (2006). Pragmatics and semantics. In L. R. Horn & G. Ward (Eds.), *The handbook of pragmatics* (pp. 442–462). Wiley-Blackwell.

RSA. (2014). *2013 a year in review* (RSA Monthly Fraud Report). Hopkinton: EMC. Retrieved from `http://www.emc.com/collateral/fraud-report/rsa-online-fraud-report-012014.pdf`

Russell, S., & Norvig, P. (2003). *Artificial intelligence: A modern approach* (2nd ed.). Upper Saddle River: Pearson Education.

Salperwyck, C., & Lemaire, V. (2011). Learning with few examples: An empirical study on leading classifiers. In *The 2011 international joint conference on neural networks (IJCNN)* (pp. 1010–1019). IEEE.

Salton, G., Wong, A., & Yang, C. (1975). A vector space model for automatic indexing. *Communications of the ACM*, *18*(11), 613–620.

Sangani, K. (2003, September). The battle against identity theft. *The Banker*, *153*(931), 180.

Sarika, S., & Varghese, P. (2013, May). Distributed software agents for antiphishing. *International Journal of Computer Science Issues*, *10*(3), 125–130.

Schank, R. C. (1999). *Dynamic memory revisited.* Cambridge, MA: Cambridge University Press.

Schank, R. C., & Abelson, R. C. (1977). *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures.* Hillsdale: Lawrence Eribaum Associates.

Segura, J. (2015, January 21). *Exploit kits: A fast growing threat.* Retrieved 2016-06-17, from `https://blog.malwarebytes.com/101/2015/01/exploit-kits-a-fast-growing-threat/`

Shimomura, T. (1996). *Takedown: The pursuit and capture of Kevin Mitnick, America's most wanted computer outlaw-by the man who did it.* New York: Hyperion.

Shreeram, V., Suban, M., Shanthi, P., & Manjula, K. (2010). Anti-phishing detection of phishing attacks using genetic algorithm. In *Communication control and computing technologies (icccct), 2010 ieee international conference on* (pp. 447–450).

Sowa, J. F. (2000). *Knowledge representation: Logical, philosophical, and computational foundations.* Pacific Grove: Brooks/Cole.

SpamAssassin. (2004). *SpamAssassin public mail corpus.* Corpus. Retrieved from `https://spamassassin.apache.org/publiccorpus/`

Sterling, B. (1993). *The hacker crackdown: Law and disorder on the electronic frontier.* New York: Bantam Books.

Stock, B., Livshits, B., & Zorn, B. (2015). *KIZZLE: A signature compiler for exploit kits* (Tehnical Report). Redmond: Microsoft. Online. Retrieved 2016-06-17, from `http://research.microsoft.com/pubs/240495/tr.pdf`

Sun, D. (2016, May 20). Hackers demand ransom payment from Kansas Heart Hospital for files. *KWCH*. Retrieved 2016-06-06, from `http://www.kwch.com/content/news/Hackers-demand-ransom-payment-from-Kansas-Heart-Hospital-380342701.html`

Taylor, J. M. (2010). Computational semantic detection of information overlap in text. In *Cognitive science conference* (pp. 2170–2175).

Tirpak, J. A. (2000, July). Find, fix, track, target, engage, assess. *The Air Force Magazine*. Retrieved 2016-06-16, from `http://www.airforcemag.com/MagazineArchive/pages/2000/july%202000/0700find.aspx`

Touretzky, D. S. (1986). *The mathematics of inheritance systems.* Pitman Publishing Limited.

Tout, H., & Hafner, W. (2009). Phishpin: An identity-based anti-phishing approach. In *2009 international conference on computational science and engineering* (Vol. 3, pp. 347–352).

TrustedSec. (2013, September 11). *Introducing SpearPhisher a simple phishing email generation tool.* Retrieved from `https://www.trustedsec.com/september-2013/introducing-spearphisher-simple-phishing-email-generation-tool/`

Tsipenyuk, K., Chess, B., & McGraw, G. (2005). Seven pernicious kingdoms: A taxonomy of software security errors. *Security & Privacy, IEEE*, *3*(6), 81–84.

Tufte, E. R. (1990). *Envisioning information.* Chesire: Graphics Press LLC.

U.S. Department of State. (2014). *Virtual reading room documents search results.* Search results. Retrieved from `https://foia.state.gov/Search/Results.aspx?collection=Clinton%5fEmail`

Weapon X. (1996, January 28). *AOL for free?* [Newsgroup]. Newsgroup. Retrieved from `https://groups.google.com/d/forum/alt.2600`

Winton, R. (2016, February 18). Hollywood hospital pays $17,000 in bitcoin to hackers; FBI investigating. *LA Times*. Retrieved 2016-06-06, from `http://www.latimes.com/business/technology/la-me-ln-hollywood-hospital-bitcoin-20160217-story.html`

Witten, I. H., Eibe, F., & Hall, M. A. (2011). *Data mining: practical machine learning tools and techniques.* Burlington, MA: Morgan Kaufmann.

Wozniak, S., & Smith, G. (2006). *iWoz : computer geek to cult icon : how I invented the personal computer, co-founded Apple, and had fun doing it.* New York: W. W. Norton.

Wu, M., Miller, R. C., & Little, G. (2006). Web Wallet: preventing phishing attacks by revealing user intentions. In *Proceedings of the second symposium on usable privacy and security* (pp. 102–113). ACM.

Wu, Y. (2005, June 27). *Email messages corpus parsed from W3C lists (for TRECENT 2005).* Corpus. Retrieved from `http://tides.umiacs.umd.edu/webtrec/trecent/parsed%5fw3c%5fcorpus.html`

Xiang, G., Hong, J., Rose, C. P., & Cranor, L. (2011, September). CANTINA+: A feature-rich machine learning framework for detecting phishing web sites. *ACM Transactions on Information and System Security*, *14*, 1–28.

Yu, W. D., Nargundkar, S., & Tiruthani, N. (2009). PhishCatch – a phishing detection tool. In *33rd annual ieee international computer software and applications conferenc* (pp. 451–456). IEEE.

Yue, Z., Hong, J. L., & Cranor, L. F. (2007). CANTINA: a content-based approach to detecting phishing web sites. *Proceedings of the 16th international conference on World Wide Web*, 639–648.

Yus, F. (2011). *Cyberpragmatics.* Amsterdam: John Benjamins Publishing Company.

VITA

VITA

Courtney Allen Falk was born August 15, 1980 in Honolulu, Hawaii. He earned his two preceding higher education degrees from Purdue University; a bachelor of science in computer science in 2003 and a master of arts in information security (granted by the philosophy department) in 2005. He has worked both for the federal government and for private industry as an information security expert. Courtney remains interested in interdisciplinary research to assist in the understanding and improvement of information security.