TA-COS 2016
Text Analytics for Cybersecurity and Online Safety

23 May 2016

PROCEEDINGS

Editors:

Guy De Pauw, Ben Verhoeven, Bart Desmet, Els Lefever

Proceedings of the LREC 2016 Workshop
"TA-COS 2016 – Text Analytics for Cybersecurity and Online Safety"

23 May 2016 – Portorož, Slovenia

Editors: Guy De Pauw, Ben Verhoeven, Bart Desmet, Els Lefever

http://www.ta-cos.org

Acknowledgments: this workshop was made possible with the support of the AMiCA (IWT SBO-project 120007) project, funded by the Flemish government agency for Innovation by Science and Technology (IWT).

Workshop Programme

14:00 - 15:00 – Keynote

Anna Vartapetiance and Lee Gillam, *Protecting the Vulnerable: Detection and Prevention of Online Grooming*

15:00 - 16:00 – Workshop Papers I

Haji Mohammad Saleem, Kelly P Dillon, Susan Benesch and Derek Ruths, *A Web of Hate: Tackling Hateful Speech in Online Social Spaces*

Stéphan Tulkens, Lisa Hilte, Elise Lodewyckx, Ben Verhoeven and Walter Daelemans, *A Dictionary-based Approach to Racism Detection in Dutch Social Media*

16:00 - 16:30 – Coffee break

16:30 - 18:00 – Workshop Papers II

Adeola O Opesade, Mutawakilu A Tiamiyu and Tunde Adegbola, *Forensic Investigation of Linguistic Sources of Electronic Scam Mail: A Statistical Language Modelling Approach*

Richard Killam, Paul Cook, Natalia Stakhanova, *Android Malware Classification through Analysis of String Literals*

Shomir Wilson, Florian Schaub, Aswarth Dara, Sushain K. Cherivirala, Sebastian Zimmeck, Mads Schaarup Andersen, Pedro Giovanni Leon, Eduard Hovy and Norman Sadeh, *Demystifying Privacy Policies with Language Technologies: Progress and Challenges*

Workshop Organizers

Guy De Pauw	CLiPS - University of Antwerp, Belgium
Ben Verhoeven	CLiPS - University of Antwerp, Belgium
Bart Desmet	LT3 - Ghent University, Belgium
Els Lefever	LT3 - Ghent University, Belgium

Workshop Programme Committee

Walter Daelemans (chair)	CLiPS - University of Antwerp, Belgium
Veronique Hoste (chair)	LT3 - Ghent University, Belgium
Fabio Crestani	University of Lugano, Switserland
Maral Dadvar	Twente University, The Netherlands
Lee Gillam	University of Surrey, UK
Chris Emmery	University of Antwerp, Belgium
Giacomo Inches	Fincons Group AG, Switzerland
Eva Lievens	Ghent University, Belgium
Shervin Malmasi	Harvard Medical School, USA
Nick Pendar	Skytree Inc, USA
Karolien Poels	University of Antwerp, Belgium
Awais Rashid	Lancaster University, UK
Cynthia Van Hee	Ghent University, Belgium
Anna Vartapetiance	University of Surrey, UK

Table of Contents

Author Index

Preface

Text analytics technologies are being widely used as components in Big Data applications, allowing for the extraction of different types of information from large volumes of text. A growing number of research efforts is now investigating the applicability of these techniques for cybersecurity purposes. Many applications are using text analytics techniques to provide a safer online experience, by detecting unwanted content and behavior on the Internet. Other text analytics approaches attempt to detect illegal activity on online networks or monitor social media against the background of real-life threats. Alongside this quest, many ethical concerns arise, such as privacy issues and the potential abuse of such technology. The first workshop on Text Analytics for Cybersecurity and Online Safety (TA-COS 2016) aims to bring together researchers that have an active interest in the development and application of such tools.

Following our call for papers, we received papers on a wide range of topics and with the help of our varied team of reviewers were able to select the most relevant and most interesting contributions. The first two papers that are presented at this workshop deal with the issue of identifying hate speech on social media. Saleem and colleagues describe a technique that automatically detects hateful communities, while Tulkens *et al.* present research on how to develop techniques that idenfity hateful words and phrases on social media.

In the second session of this workshop Opesada *et al.* present a study on the origin of 419 Scam e-mails, using text classification techniques that identify varieties of English. Killam *et al.* identify malware on the Android platform by using text analytics on the apps' binary files. Finally, Wilson *et al.* describe work on developing techniques that can aid people in understanding the often lengthy and complex terms of use that they agree to online.

We are very pleased with this wide variety of topics of the submitted papers and are furthermore very pleased to be able to kick off our workshop with a keynote lecture by Anna Vartapetiance of the University of Surrey's Centre for Cyber Security. She will present ongoing research on the automatic detection of online grooming. We are sure that the presentations at TA-COS 2016 will trigger fruitful discussions and will help foster the awareness of the increasingly important role text analytics can play in cybersecurity applications.

The TA-COS 2016 Organizers,
Guy De Pauw
Ben Verhoeven
Bart Desmet
Els Lefever
`www.ta-cos.org`

A Web of Hate: Tackling Hateful Speech in Online Social Spaces

Haji Mohammad Saleem[1], **Kelly P Dillon**[2], **Susan Benesch**[3], **and Derek Ruths**[1]
[1]School of Computer Science, McGill University, Montreal
[2]School of Communication, The Ohio State University, Ohio
[3]Berkman Center for Internet & Society, Harvard University, Massachusetts
haji.saleem@mail.mcgill.ca, dillon.148@osu.edu,sbenesch@cyber.law.harvard.edu, derek.ruths@mcgill.ca

Abstract

Online social platforms are beset with hateful speech - content that expresses hatred for a person or group of people. Such content can frighten, intimidate, or silence platform users, and some of it can inspire other users to commit violence. Despite widespread recognition of the problems posed by such content, reliable solutions even for detecting hateful speech are lacking. In the present work, we establish why keyword-based methods are insufficient for detection. We then propose an approach to detecting hateful speech that uses content produced by self-identifying hateful communities as training data. Our approach bypasses the expensive annotation process often required to train keyword systems and performs well across several established platforms, making substantial improvements over current state-of-the-art approaches.

Keywords: Hate speech, social media, text classification

1. Introduction

Online spaces are often exploited and misused to spread content that can be degrading, abusive, or otherwise harmful to people. An important and elusive form of such language is *hateful speech*: content that expresses hatred of a group in society.

Hateful speech has become a major problem for every kind of online platform where user-generated content appears: from the comment sections of news websites to real-time chat sessions in immersive games. Such content can alienate users and can also support radicalization and incite violence (Allan, 2013). Platform operators recognize that hateful content poses both practical and ethical issues and many, including Twitter, Facebook, Reddit, and gaming companies such as Riot Games, have tried to discourage it, by altered their platforms or policies.

Yet reliable solutions for online hateful speech are lacking. Currently, platforms predominantly rely on users to report objectionable content. This requires labor-intensive review by platform staff and can also entirely miss hateful or harmful speech that is not reported. With the high volume of content being generated on major platforms, an accurate automated method might be a useful step towards diminishing the effects of hateful speech.

Without exception, state-of-the-art computational approaches rely upon either human annotation or manually curated lists of offensive terms to train classifiers (Kwok and Wang, 2013; Ting et al., 2013). Recent work has shown that human annotators tasked with labeling hate speech have significant difficulty achieving reasonable inter-coder reliability (Kwok and Wang, 2013). Within industry, it is generally acknowledged that keyword lists are also insufficient for accurate detection of hateful speech. However, little work has been done to understand the nature of their limitations and to design able alternative approaches. This is the topic of the present work.

This paper makes three key contributions. First, we establish why the problem of hateful speech detection is difficult, identifying factors that lead to the poor performance of keyword-based approaches. Second, we propose a new approach to hateful speech detection, leveraging online communities as a source of language models. Third, we show that such a model can perform well both within a platform and *across* platforms — a feature we believe we are the first to achieve.

We are also aware that automated detection of online speech could be misused to suppress constructive and/or dissenting voices by directing the system at individuals or groups that are not dedicated to expressing hatred. Such a use would be antithetical to our intent, which is to explore and illustrate ways in which computational techniques can provide opportunities to observe and contain harmful content online, without impinging on the freedom to speak openly, and even to express unpalatable or unpopular views. We hope that our work can help diminish hatred and harm online. Furthermore, since our method can be trained on and applied to a wide array of online platforms, this work may help to inform the direction of future research in this area.

2. Background

Hate and hateful speech. Legal and academic literature generally defines hate speech as speech (or any form of expression) that expresses (or seeks to promote, or has the capacity to increase) hatred against a person or group of people because of a characteristic they share, or a group to which they belong (Mendel et al., 2012). There is no consensus definition, however. Definitions of this sort are problematic for a number of reasons (Bartlett et al., 2014), including that hate speech is defined by prevailing social norms, context, and individual and collective interpretation. This makes it difficult to identify hate speech consistently and yields the paradox (also observed with pornography) that each person seems to have an intuition for what hate speech is, but rarely are two people's understandings the same. This claim is affirmed by a recent study that demonstrated a mere 33% agreement between coders from differ-

ent races, when tasked to identify racist tweets (Kwok and Wang, 2013).

A particular ambiguity in the term 'hate speech' is in "hate" itself. That word might refer to the speaker/author's hatred, or his/her desire to make the targets of the speech feel hated, or desire to make others hate the target(s), or the apparent capacity of the speech to increase hatred. Needless to say, we require a rigorous — and formal — definition of a type of speech if we are to automate its detection.

Our initial motivation was to find, and work with, a notion of hate speech that can be operationalised. The work of online platform operators (e.g., Twitter, Facebook, and Reddit) helped to focus this aim. Their concern over the capacity of language to do harm — whether emotional, mental, or physical — logically focuses more on what is *expressed* rather than how it is *intended*. Whereas "hate speech" can imply an inquiry or judgment about intent (e.g. what was this person feeling or wishing?), we propose the term "hateful speech" to focus on the expression of hate — a nuanced, but useful distinction since expression is easier to detect than intent, and more likely to be linked to language's capacity to cause harm.

This leads to our term *hateful speech*: speech which contains an expression of hatred on the part of the speaker/author, against a person or people, based on their group identity.

Hateful speech is not to be mistaken for "cyber-bullying," another form of troubling online content that has been widely discussed and studied in recent literature. Cyberbullying is repetitive, intentional, aggressive behavior against an individual, and it either creates or maintains a power imbalance between aggressor and target (Tokunaga, 2010). It is often hateful but it does not necessarily denigrate a person based on his or her membership in a particular group, as hateful speech (the subject of the present work) does.

Community-defined speech. As we will discuss in detail later, we use the language that emerges from self-organized communities (in Reddit and elsewhere) as the basis for our models of hateful speech. Our decision is based on a deep sociological literature that acknowledges that communities both form, and are formed by, coherent linguistic practices (Bucholtz and Hall, 2005). Most groups are defined in part by the "relationships between language choice and rules of social appropriateness" forming speech communities (Gumperz, 2009). In this way of thinking, the group is defined by the speech and the speech comes to define the group (Klein et al., 2007; Reicher et al., 1995; Spears and Lea, 1992; Spears and Lea, 1994).

In the context of this study, this means that hate groups and the hateful speech they deploy towards their target community cannot exist without one another, especially online. Therefore, taking the linguistic attributes particular to a community committed to degrading a specific group is a legitimate and principled way of defining a particular form of hateful speech. To our knowledge, this work represents the first effort to explicitly leverage a community-based classification of hateful language.

Existing approaches to detecting hateful speech. Despite widespread concern about hateful speech online, to our knowledge there have been only three distinct lines of work on the problem of automated detection of hateful speech. One study concerned the detection of racism using a Naive Bayes classifier (Kwok and Wang, 2013). This work established the definitional challenge of hate speech by showing annotators could agree only 33% of the time on texts purported to contain hate speech. Another considered the problem of detecting anti-Semitic comments in Yahoo news groups using support vector machines (Warner and Hirschberg, 2012). Notably, the training data for this classifier was hand-coded. As we will discuss in this paper, manually annotated training data admits the potential for hard-to-trace bias in the speech ultimately detected. A third study used a linguistic rule-based approach on tweets that had been collected using offensive keywords (Xiang et al., 2012). Like manually annotated data, keyword-based data has significant biasing effects as well.

In this work we aim to build on these studies in two ways. First, we will consider a definition of hateful speech that could be practically useful to platform operators. Second, we will develop a general method for the detection of hateful speech that does not depend on manually annotated or keyword-collected data.

Reddit and other online sources of hateful speech. Reddit is currently one of the most actively used social content aggregation platforms. It is used for entertainment, news and social discussions. Registered users can post and comment on content in relevant community discussion spaces called *subreddits*. While the vast majority of content that passes through Reddit is civil, multiple subreddits have emerged with the explicit purpose of posting and sharing hateful content, for example, `r/CoonTown`, `r/FatPeopleHate`, `r/beatingwomen`; all which have been recently banned under Reddit's user-harassment policy (Moreno et al., 2015). There are also subreddits dedicated to supporting communities that are the targets of hate speech.

Reddit is an attractive testbed for work on hateful speech both because the community spaces are well-defined (i.e., they have names, complete histories of threaded discussions) and because, until recently, Reddit has been a major online home for both hateful speech communities and supporters for their target groups. For these reasons, throughout this paper, our analyses heavily leverage data from Reddit groups.

Of course, Reddit is not the sole platform for hateful speech. Voat, a recently created competitor to Reddit, along with a vibrant ecosystem of other social content aggregation platforms, provide online spaces for topical discussion communities, hate groups among them. Furthermore, dedicated websites and social networking sites such as Twitter and Facebook are also reservoirs of easily accessible hateful speech.

Important research has investigated the effects of racist speech (Nakamura, 2009) and sexual harassment (Fox and Tang, 2014) in online games. Notably, in this study we have not worked with data from online gaming platforms, primarily because the platforms are generally closed to con-

Target Group	Hate subreddit	# of comments	Support subreddit	# of comments
Black	CoonTown	350851	Racism	9778
Plus	FPH	1577681	LoseIt	658515
Female	TRP	51504	TwoXCr	66390

Table 1: Public comments collected from hate and support subreddits on Reddit, for three target groups. (*FPH: FatPeopleHate, TRP: TheRedPill, TwoXcr: TwoXChromosomes*)

ventional data collection methods.

3. The limits of keyword-based approaches

In the same way that hateful groups have defining speech patterns, communities that consist of the targets of hateful speech also have characteristic language conventions. We will loosely call these *support groups*. Notably, support groups and the groups that espouse hateful speech about them often engage in discourse on similar topics, albeit with very different intent. Fat-shaming groups and plus-size communities both discuss issues associated with high BMI, and women and misogynists both discuss gender equity. This topical overlap can create opportunities for shared vocabulary that may confuse classifiers.

In addition, many keyword-based approaches select established and widely known slurs and offensive terms that are used to target specific groups. While such keywords will certainly catch some hateful speech, it is common to express hate in less explicit terms, without resorting to standard slurs and other offensive terms.

For example, hateful speakers refer to migrants and refugees as "parasites" and call African-Americans "animals." While neither of these terms are inherently hateful, in context they strongly denigrate the group to which each term is applied.

We can expect that classifiers trained on overtly hateful keywords will miss such posts that use more nuanced or context-dependent ways of achieving hateful speech.

Furthermore, keywords can be also be obscured through misspelings, character substitutions (by using symbols as letters), using homophones etc. These practices are commonly employed to circumvent keyword-based filters on online platforms (Warner and Hirschberg, 2012).

In this section, we study the potential impact of topic overlap on data returned by keyword-based queries (we will consider under-sampling issues in the next section). Here our focus will be on the sample that keyword-based filters return and in later sections we will consider the performance of classifiers built from such samples.

Data. Recently, Reddit user, `Stuck_In_the_Matrix`[1], made available large data dumps that contain a majority of the content (posts and comments) generated on Reddit[2]. The data dumps, collected using the Reddit API, are organized by month and year. The data date back to 2006 and are regularly updated with new content. We use all comments from January 2006 through January 31, 2016 and expanded the dataset with each update. Each file corresponds to a month of Reddit data, and every line is a json object of a Reddit comment or post.

For our analysis, we identify three commonly targeted groups on Reddit — African-American (black), plus-sized (plus) and women. For each of the target groups, we select the most active support and hate subreddits. To create our datasets, we extract all user comments in the selected subreddits from the data dumps described above, in October 2015. The details on the selected subreddits and the number of the extracted comments are provided in Table 1.

Black		Plus-size		Female	
Coon-Town	racism	FPH	loseit	TRP	TwoXCr
nigger	**white**	**weight**	**weight**	**women**	**time**
white	racism	**calorie**	**calorie**	**girl**	**women**
black	**black**	**time**	**time**	**time**	**feel**
shit	**racist**	**work**	**food**	woman	**work**
time	**race**	**food**	**eating**	**shit**	**year**
fucking	**time**	**feel**	**week**	**work**	**fuck**
fuck	person	**eating**	**work**	**year**	**shit**
race	point	**week**	**feel**	**life**	weight
year	feel	**lose**	**lose**	**fuck**	**fucking**
hate	comment	**year**	**diet**	guy	person
racist	american	women	**body**	point	**life**
live	post	**diet**	exercise	friend	**girl**
work	issue	**body**	**goal**	post	love
jew	asian	start	loss	**feel**	pretty
crime	color	**goal**	**year**	**fucking**	food
Jaccard Index: 0.28		JI: 0.76		JI: 0.50	

Table 2: Top discovered topics from support and hate subreddits for the three targets. The bold terms signify those that are present in both the hate and support vocabulary.

Methods. For each of the selected subreddits, we use labeled Latent Dirichlet Allocation (LLDA) to learn the topics that characterize them, against a baseline Reddit language. This baseline is intended to push the LLDA to remove non-topical vocabulary from the two subreddit topics; it consists of a sample of 460,000 comments taken at random from the Reddit data scrape (none of the posts belonged to any of the subreddits of interest). Prior to topic modeling, stop words, punctuation, URLs, and digits were stripped from the comments and for the purpose of balanced analysis, an equal number of comments was selected from the subreddit and the random sample. We use JGibbLDA for the topic inference (Phan and Nguyen, 2006).

Results. In Table 2, we present the 15 most topical words from each subreddit. The top terms in the topics are consistent with the target/support communities. For example, the term "women" was ranked highly in subreddits that concern women (whether positively or negatively referenced) and "weight" is the highest ranked topic for subreddits discussing plus-sized individuals and lifestyle.

We observe a substantial overlap in vocabulary of hate and support subreddits, across all three target communities (see bold words in Table 2). While in the case of a black target

[1] https://www.reddit.com/user/Stuck_In_the_Matrix/

[2] http://couch.whatbox.ca:36975/reddit/

group, we observe a Jaccard Index (*JI*) of 0.28, the overlap is higher in the case of female targets with *JI* at 0.50 and much higher for plus-size targets, with a *JI* of 0.76.

The implication of this shared vocabulary is that while keywords can be used to detect text relevant to the target, they are not optimal for detecting targeted hateful speech. Shared vocabulary increases the likelihood of tagging content that is related to the target but not necessarily hateful, as hateful and increases false positives. We therefore require more robust training data.

4. A community-driven model of hateful speech

A key objective of our research is to avoid the issues associated with using manual annotation and keyword searches to produce training data for a classifier. As noted previously, sociological literature acknowledges that communities are formed by coherent linguistic practices and are defined, in part, by their linguistic identity (Gumperz, 2009). Thus, the opportunity considered here is to leverage the linguistic practices of specific online communities to empirically define a particular kind of hateful speech.

Since linguistic practices coincide with the identity of a community using them, we can define hateful speech as discourse practiced by communities who self-identify as hateful towards a target group. The members of the community contribute to the denigration of the target and, therefore, share a common linguistic identity. This allows us to develop a language model of hateful speech directly from the linguistic conventions of that community without requiring manual annotation of specific passages or keyword-based searches. This approach has a number of advantages over these practices.

First, a community-based definition removes the interpretive challenge involved in manual annotation. Membership in a self-organized community that is committed to denigration of a target group through the hatred of others is an observable attribute we can use to surface hateful speech events.

Second, unlike prior work, our method does not require a keyword list. We identify communities that conform to the linguistic identity of a self-organized hateful groups and use such communities to collect data. This data is used to learn the language model around the linguistic identity for detection. This removes any biases implicit in the construction of a keyword list (i.e., in the words included in or excluded from the list).

Third, a community-based definition provides a large volume of high quality, current, labeled data for training and then subsequent testing of classifiers. Such large datasets have traditionally been difficult to collect due to dependence on either manual annotation (annotation is slow and costly) or keyword searches (stringent keywords may turn up relatively few hits).

This approach generalizes to other online environments (such as Voat and other hateful speech-focused web forums) in which communities declare their identities, intentions, and organize their discussions. Any online (or, even, offline) communication forum in which all participants gather for the understood purpose of degrading a target group constitutes a valid source of training data.

In the following subsections, this approach is validated through three analyses. First, we demonstrate that the hate speech communities identified actually employ distinct linguistic practices: we show that our method can reliably distinguish content of a hateful speech community from the rest of Reddit. We also show that our approach substantially outperforms systems built on data collected through keywords.

Second, we show that our approach is sensitive to the linguistic differences between the language of hateful and support communities. This task is notably difficult given the results we reported above, showing that such communities share many high-frequency words.

Finally, we use our Reddit-trained classifier to detect hateful speech on other (non-Reddit) platforms: on Voat and hateful speech web forums (websites devoted to discussion threads attacking or denigrating a target community). For both, we find that our method performs better than a keyword-based baseline.

4.1. Data collection

Reddit. We use Reddit as the primary source for the hateful communities and leverage the linguistic practices of these communities to empirically define and develop language models for target-specific hateful speech. In all three of our studies, we focus on the aforementioned three target groups: black people, plus-sized individuals, and women. For each, we select the most active hateful and support subreddits and collect all the publicly available comments present in the data dumps provided by `Stuck_In_the_Matrix`. The details on the dataset are provided in Table 1. We also collect a random sample of 460,000 Reddit comments to serve as negative examples.

Voat. Voat, a content aggregator similar to Reddit, also hosts active discussion communities, called *subverses*, few of which identify as hateful. We select Voat because of its similarity to our original source[3]. Since the two websites cater to a similar user-base, the generated linguistic identities should be similar in sub-communities with similar themes. Therefore, the language model of hateful communities on Reddit should match, to an extent, with the language model of similar hateful communities on Voat.

For the three target groups, we identify hateful subverses — `v/CoonTown`, `v/fatpeoplehate` and `v/TheRedPill` — sub-communities that share their name with their counterparts on Reddit and target blacks, plus-size individuals, and women, respectively. In the absence of an API, we use web-scraping libraries to retrieve all publicly available comments posted to the selected subverses between July 2015 and January 2016. We also collect a set of 50,000 comments (from the same time period) from a random sample of subverses to serve as negative examples (Table 3).

Web forums. We also use stand-alone web forums that are dedicated to expressing hate or contempt for the target communities. These web forums are social platforms that

[3] http://thenextweb.com/insider/2015/07/09/what-is-voat-the-site-reddit-users-are-flocking-to/

Target	Subverse	Voat	Website	Comments
Black	CoonTown	3358	shitskin	3160
Plus	fatpeoplehate	31717	-	
Female	TheRedPill	478	mgtowhq	20688

Table 3: Target-relevant hateful comments collected from Voat subverses and web forums.

provide their users with discussion boards, where users can create threads under predefined topics and other users can then add comments in these threads. We, therefore, select web forums for their discussion-based communities and user-generated content. Again, due to the lack of APIs, we use, as data, comments that were collected by web-scraping libraries from numerous threads of their discussion boards during October 2015.

For the black target group, we use `Shitskin.com`: our dataset consists of 3,160 comments posted to 558 threads from three of website's boards: "Primal Instinct", "Crackin the whip!" and "Underground Railroad." For the female target group, we use `mgtowhq.com`: this dataset consists of 20688 comments posted to 4,597 threads from the "MGTOW General Discussion" board. Finally, as a source of negative examples, we use the "random" discussion board on `topix.com`: this dataset consists of nearly 21,000 comments from 2458 threads. To our knowledge, no large fat-shaming forum exists, thus we do not include this target group in this phase of the study (Table 3). All comments have posting times between July 2015 and January 2016.

4.2. Methods

Before the classification process, we preprocess all the data by eliminating URLs, stopwords, numerals and punctuations. We further lowercase the text and remove platform-relevant noise (e.g., comments from house keeping bots on Reddit like AutoModerator). The text is finally tokenized and used as input for the classification pipeline.

We use multiple machine learning algorithms to generate the language models of hateful communities. From the analysis of the prior work, we identify the commonly-used algorithms and employ them in our analysis. Specifically, we use naive Bayes (NB), support vector machines (SVM) and logistic regression (LR). We do this in order to assess the merits of our insight into using community-defined data collection.

The algorithms take as input, tokenized and preprocessed arrays of user comments along with the label of the community they belong to. We use a sparse representation of unigrams with *tfidf* weights as our feature set. In future investigation, we would like to add part of speech tags and sentiment score as features.

For performance evaluation, we use the standard measures: accuracy, precision, recall and F1-Score. We also use Cohen's κ as a measure of agreement between the observed and expected labels. κ helps in evaluating the prediction performance of classifiers by taking in account any chance agreement between the labels.

Baseline comparison. Our aim is to assess the impact of using community-based text compared with keyword-based text as training data. Due to space limitations, here we report only a logistic regression classifier trained on keyword-collected data (SVM and NB showed comparable performance).

The specific keywords used are generated from the comments collected from hateful Reddit communities. For a given target group, we generate three sets of keywords for each: (1) keywords generated between hate subreddits and a random sample of Reddit comments using LLDA, as in Section 3, (2) keywords generated between hate subreddits and a random sample of Reddit comments using χ^2 weights (χ^2I), and (3) keywords generated between hate and support subreddits using χ^2 weights (χ^2II). To generate the training datasets, we use the top 30 keywords and from a separate random sample of Reddit comments, collect samples that contain at least one of the keywords as positive samples and samples that contain no keywords as negative samples. For each keyword type and each target, we aggregate 50,000 positive and 50,000 negative samples for training.

4.3. Results and Discussion

Community language vs. hateful speech. It may seem that, by comparing classifiers on the task of detecting hateful community posts, we are equating language produced by a hateful community with hateful language. Certainly, they are not always the same. Some content is likely non-hateful chatter. One alternative for excluding such noise is manual coding of testing data. Given the existing issues with such labeled data, we avoid such manual labeling. Furthermore, a comparison of the two approaches is not fair due to the associated trade-offs. The community definition, as mentioned, relies on the assumption that all the content in a hateful community is hateful, which might not always be true. However, such an assumption allows us to generate large training datasets with relative ease. We therefore allow the presence of some noise in the training data for ease of training data generation and favouring recall. On the other hand, manual annotation promises less noisy datasets at the expense of time and resources, which limits the size of training datasets. It would be very laborious to produce datasets as large as those generated with our community approach. Also, since manual annotation relies heavily on personal perception, it can also introduce noise in the datasets. In other words, manual annotation does not allow us to generate large training sets, and also cannot provide completely noise-free data.

Another option, however, is to focus on the precision ($\frac{TP}{TP+FP}$) of the classifier. Precision indicates the classifier's ability to identify only content from the hateful community. The construction of the test datasets is such that hateful speech should only exist in the hateful community posts. Thus, a method that detects hateful content should strongly favor including only content from hateful communities — yielding high precision. Crucially, in the discussions that follow, we find that a community-based classifier demonstrates much higher precision than keyword-based methods. Thus, by either measure (F1 or precision), our community-based classifier outperforms the baselines.

(a) Assessing the distinct nature of language emerging from hate groups.

Target	Accuracy			Precision			Recall			F1-Score			Cohen's κ		
	NB	SVM	LR	NB	SVM	LR	NB	SVM	LR	NB	SVM	LR	NB	SVM	LR
Black	0.79	0.81	0.81	0.78	0.84	0.87	0.82	0.74	0.73	0.8	0.79	0.79	0.58	0.61	0.61
Plus	0.78	0.78	0.79	0.78	0.81	0.82	0.79	0.75	0.73	0.78	0.78	0.77	0.56	0.57	0.57
Female	0.77	0.8	0.81	0.71	0.81	0.84	0.9	0.77	0.75	0.79	0.79	0.79	0.55	0.6	0.61

(b) Assessing sensitivity between the language of hate and support groups.

Black	0.8	0.79	0.79	0.8	0.8	0.78	0.85	0.82	0.86	0.82	0.81	0.82	0.57	0.56	0.55
Plus	0.83	0.85	0.85	0.85	0.84	0.84	0.79	0.86	0.86	0.82	0.85	0.85	0.66	0.69	0.7
Female	0.79	0.78	0.78	0.78	0.79	0.8	0.79	0.77	0.77	0.79	0.78	0.78	0.57	0.56	0.57

Table 4: The performance of the three classification algorithms across the three target groups, with a 10 fold cross-validation. (a) Hateful comments are classified against random comments. (b) Hateful comments are classified against comments from support communities. In both cases, the classifier is able to distinguish hate speech from negative cases. (NB: Naive Bayes, SVM: Support Vector Machines, LR: Logistic Regression)

(a) Baseline performance over Reddit data.

Target	Accuracy			Precision			Recall			F1-Score			Cohen's κ		
	LDA	χ^2I	χ^2II	LDA	χ^2I	χ^2II	LDA	χ^2I	χ^2II	LDA	χ^2I	χ^2II	LDA	χ^2I	χ^2II
Black	0.59	0.63	0.57	0.61	0.71	0.62	0.52	0.44	0.4	0.56	0.54	0.48	0.18	0.26	0.15
Plus	0.53	0.57	0.53	0.54	0.6	0.55	0.35	0.4	0.34	0.42	0.48	0.42	0.06	0.14	0.06
Female	0.68	0.7	0.7	0.65	0.69	0.74	0.71	0.71	0.6	0.68	0.7	0.66	0.35	0.40	0.4

(b) Baseline performance over Voat data.

Black	0.62	0.63	0.62	0.65	0.73	0.68	0.48	0.4	0.4	0.55	0.51	0.51	0.24	0.26	0.23
Plus	0.56	0.6	0.57	0.58	0.65	0.61	0.35	0.4	0.36	0.43	0.5	0.45	0.11	0.2	0.14
Female	0.67	0.69	0.67	0.68	0.71	0.74	0.63	0.63	0.5	0.65	0.67	0.6	0.35	0.38	0.34

(c) Baseline performance over web forum data.

Black	0.66	0.62	0.57	0.72	0.77	0.67	0.53	0.35	0.31	0.61	0.48	0.42	0.32	0.24	0.15
Female	0.78	0.79	0.77	0.81	0.83	0.87	0.75	0.74	0.64	0.78	0.78	0.74	0.56	0.58	0.54

Table 5: We calculate the baseline performance on multiple platforms with three keyword-generating methods: LDA, χ^2I and χ^2II. Classification was done using logistic regression.

Target	Acc	Pre	Rec	F1	κ
Voat					
Black	0.82	0.87	0.74	0.80	0.64
Plus	0.81	0.85	0.74	0.79	0.62
Female	0.74	0.76	0.71	0.73	0.49
Websites					
Black	0.82	0.87	0.77	0.82	0.65
Female	0.77	0.83	0.69	0.75	0.54

Table 6: For our targets, we collect comments from hateful communities on Voat and web forums and test the performance of language models learned from Reddit communities.

Training	Testing	Acc	Pre	Rec	F1	κ
CT	FPH	0.58	0.72	0.26	0.38	0.15
CT	TRP	0.55	0.6	0.22	0.32	0.08
FPH	TRP	0.58	0.65	0.3	0.41	0.15
FPH	CT	0.54	0.61	0.23	0.34	0.08
TRP	CT	0.51	0.53	0.28	0.36	0.03
TRP	FPH	0.6	0.65	0.41	0.51	0.19

Table 7: We test the performance of classification systems built on data that belongs to a target community differnt than the one we test on. (CT: CoonTown)

Hateful groups have distinct linguistic signatures. In Table 4(a), we see the performance of the three classifiers when classifying a balanced corpus of hateful posts and randomly selected (non-hateful speech) Reddit posts with 10-fold cross validation. The dataset consists of all the comments collected from the relevant hate subreddit (Table 1) as positive samples and an equal number of random Reddit comments as negative samples. We observe the three classifiers perform almost identically. Naive Bayes slightly outperforms others on Recall and F1-score, while Logistic Regression is a slightly better performer on the other metrics. Also, the performance of the classifiers is consistent across the three target groups. Analysis of κ suggests that observed labels after the classification process are in moderate to substantial agreement with the expected labels.

Comparison to baseline. In all cases considered, a classifier trained on community-based data outperforms a keyword-based classifier. Notably, the keyword-based clas-

sifier for the women-target group performed best, suggesting that hateful community language associated with the keywords used for collection are more representative of hateful speech (compared to other communities).
From a precision perspective, we find that the community-based classifier outperforms the baselines by between 10% and 20%, indicating that the community-based classifier is including far fewer incorrect cases of hateful speech (false positives). When we look at the true positive posts that have been detected exclusively by the community-based classifier (i.e., that the keyword-based approach missed), we find many that are clearly hateful, but in ways that do not use specialized slurs. Several examples from the `CoonTown` subreddit:

1. "I don't see the problem here. Animals attack other animals all the time."
2. "Oy vey my grandparents vuz gassed ven dey vaz six years old!"
3. "DNA is rayciss, or didn't you know?"
4. "Are they going to burn their own town again? Yawn."

These examples characterize different (and important) ways in which speech can be hateful without using words that typically operate, largely independent of context, as slurs. In Example 1, African-Americans are described as animals, employing a word that is not usually a slur, to denigrate them. In Example 2, historical context (the gas chambers in Nazi concentration camps), culturally stereotyped language ("Oy vey"), and spelling to imitate an accent ("ven dey vaz") are successfully used to express contempt and hatred, without any slur or even any word that, like 'animals' in the first example, is sometimes pressed into service as a slur. The third example, like the second, parodies an accent, and here it is notable that while "racist" might be a keyword use for collection, it's unlikely that "rayciss" would be used. Finally Example 4 achieves its effect by attacking a group through an implication of stereotyped action without even actually naming them at all (as opposed to Example 1, in which the targets were called "animals").

Community-based approach is sensitive to the linguistic differences of hate and support communities. In Section 3, we showed that hateful and support communities for a target group have a shared vocabulary: the two communities often engage in discourse on similar topics, albeit with quite different intent. Since the shared keywords are not effective in the discrimination process, recognizing the distinction between hate and support communities can be challenging. We set up a classification task for identifying comments from support and hate communities, carried out with a 10-fold cross-validation. The performance of the task is presented in Table 4(b). We observe that this performance is close to the performance of our system against a random collection of Reddit comments (Table 4(a)). Therefore, even with shared vocabulary, our system is sensitive to the distinction in linguistic characteristics of hateful and support communities for the same target.

Community-trained systems can be deployed on other platforms. Often training data for hateful language classification can be hard to obtain on specific platforms. For this reason, methods that work across platforms (trained on one platform, applied on another platform) present significant advantages.
For the analysis, we continue with the same three target groups and train our language model, using logistic regression, with comments from relevant Reddit communities and then test it on data we collected from other platforms. The performance of the system, (Table 6), is very similar to the results we obtain when testing on Reddit (Table 4(a)). This said, we must be careful not to overstate our method's generalizability. While, certainly, the degree of generalizability observed is noteworthy (particularly given past work), these platforms all feature similar posting conventions: posts are not length restricted, are made within well defined discussion threads, and have a clear textual context. Our method will likely perform well on any such forum-based system. Platforms, which involve quite different conventions, particularly those that are predominantly populated by short-text posts (e.g., Twitter and Facebook), will likely involve additional work. Nonetheless, we do believe that the community-based approach presents opportunities for these other platforms as well.

Hateful classifiers are not target-independent. Hateful conversations are thematic and major topics discovered from conversations are target related (Table 2). Not surprisingly, our system performs poorly when tested across targets. We train the classifier on one target and test it on another. The results (see Table 7) provide a strong indication that hateful speech classification systems require target-relevant training.

Detailed Error Analysis. In order to better understand the performance of our system, we manually inspect a set of erroneously classified posts from the *coontown* training/testing dataset. We characterize the kinds of issues we observe and discuss them here.
Type I errors. These posts arise when non-hate group posts are labeled as hate-group posts. Notably, we observe that some of these errors are actually racist comments that originated from other communities in Reddit.

1. "well jeez if u pit a nigger against a cunt what do u expect"
2. "Triskaid is a fucking nigger."

In both of the cases the comments were in fact racist and were therefore correctly labeled. This, of course, points out a potential (though, we would argue minor) weakness of our approach, which is that hate groups are not the *only* source of hateful language — simply the most high-density source.
More frequently, Type I errors featured non-racist comments which had been mislabeled. This is likely due to the fact that not all content in a hateful community is hateful: some is simply off-topic banter among community members. This adds noise during the training phase which manifests as classification errors. While certainly an issue, given

the dramatic improvement in overall classification performance, we consider this an acceptable trade off at this stage in the research. Future work should consider ways of focusing training data further on the distinctly hateful content produced by these communities.

Type II errors. In most cases where hateful-speech community posts were incorrectly labeled as non-hateful, we primarily find that these were, in fact, non-racist posts that were made to the hateful subreddit. Here are a few examples:

1. "and you're a pale virgin with a vitamin d deficiency."
2. "Whats the deal with you 2? And besides, we're all on the same side here.."
3. "IP bans do literally nothing, it only takes a moment to change it."
4. "I can't believe Digg is still up. I can't believe Reddit is still up."

Posts like these constitute noise, in terms of our community-based definition of hateful speech, discussed above. Nonetheless, our system was able to correctly identify them as non-hateful. Taken together with the Type I errors, it appears that the noise implicit in our community-definition of hateful speech yields a modest increase in Type I error, but can somewhat be removed by the classifier in the form of Type II errors (which are not, in fact, errors). A very small number of other Type II errors are examples of hateful speech, but that target a community other than blacks (in the cases we saw, primarily Jews):

1. "Peace and harmony? Yeah that's why they stole that land (now kikeriel) and killed the civilians that lived there before. Did I mention they STILL kill the Palestinians to this day and cover it up? Fuck them."
2. "quit kissing kikeass"
3. "You sound like a jew. In a system ruled by money, money can buy anything. Everything is capitalisms fault. But I get why you'd support capitalism since your "people" invented the whole shebang"
4. "Losing weight isn't even hard, stop eating like a fucking landwhale, drink lots of water and move your fatass"

Although these comments are hateful, since they are not directed at black people, the system is technically performing according to specification.

Our system missed some cases of obvious racism, such as the following examples. However, such cases constitute only a small fraction of the comments in Type II error.

1. "Ok Korea - you know your duty in the impending 'blackification' of the globe? I know where I stand"
2. "Black people are terrible. "
3. "Pretty soon we will need a dedicated sub for black-on-senior sexual assaults."
4. "Who is the target audience? I would think black literacy levels would prevent "nig lit" from ever being a viable book market."

Overall, our analysis of Type II errors indicated that the vast majority of mislabeled comments are not racist and are, therefore, correctly labeled. This suggests that the actual performance of our method is likely higher than what we report.

Imbalanced Datasets We use balanced datasets for our analysis. Since this assumption may or may not hold for different data sources, we perform some initial analysis on imbalanced datasets. As the actual composition of data sources can be variable, we generate testing sets with the ratio of hateful content to non-hateful content at 1:10, 1:100, 1:1000. Our preliminary results are similar to the performance on a balanced test set. These results are encouraging but require further analysis. We hope to overcome the challenges of dataset-shift due to mismatch in the composition of testing and training datasets in future work.

5. Conclusion

The presence of hateful speech on online platforms is a growing problem with a need for robust and scalable solutions. In this work, we investigated the limitations of keyword-based methods and introduced a community-based training method as an alternative. Our work makes two key contributions.

First, we highlight two major mechanisms that hurt the performance of keyword-based methods. The shared vocabulary between hateful and support communities causes training positive examples to contain non-hateful content. Also, because keyword lists focus on more widely known slurs, these lists miss many instances of hateful speech that use less common or more nuanced constructions to express hatred all too clearly.

Our second contribution is the idea of using self-identified hateful communities as training data for hateful speech classifiers. This approach both involves far less effort in collecting training data and also produces superior classifiers.

The promising results obtained in this study suggest several opportunities for future work. Foremost is the extension of this approach to other non-forum-based platforms. Twitter and Facebook, for example, are heavily used platforms which mainly feature short-text messages. Such content presents unique challenges that will require new or modified approaches. Another direction involves looking at other high-signal features (syntax, n-grams, and sentiment scores).

In these and other initiatives, we believe that community-based data may play an essential role in producing both better detectors of hateful speech, and a richer understanding of the underlying phenomenon.

6. Bibliographical References

Allan, J. (2013). The harm in hate speech. *Constitutional Commentary*, 29(1):59–80.

Bartlett, J., Reffin, J., Rumball, N., and Williamson, S. (2014). Anti-social media. *Demos*, pages 1–51.

Bucholtz, M. and Hall, K. (2005). Identity and interaction: A sociocultural linguistic approach. *Discourse studies*, 7(4-5):585–614.

Fox, J. and Tang, W. Y. (2014). Sexism in online video games: The role of conformity to masculine norms and social dominance orientation. *Computers in Human Behavior*, 33:314–320.

Gumperz, J. J. (2009). The speech community. *Linguistic anthropology: A reader*, 1:66.

Klein, O., Spears, R., and Reicher, S. (2007). Social identity performance: Extending the strategic side of side. *Personality and Social Psychology Review*, 11(1):28–45.

Kwok, I. and Wang, Y. (2013). Locate the hate: Detecting tweets against blacks. In *AAAI*.

Mendel, T., Herz, M., and Molnar, P. (2012). Does international law provide for consistent rules on hate speech? *The content and context of hate speech: Rethinking regulation and responses*, pages 417–429.

Moreno, J., Pao, E., and Ohanian, A. (2015). Removing harassing subreddits. *Reddit*, June. Retrieved from https://www.reddit.com/r/announcements/comments/39bpam/removing_harassing_subreddits/.

Nakamura, L. (2009). Don't hate the player, hate the game: The racialization of labor in world of warcraft. *Critical Studies in Media Communication*, 26(2):128–144.

Phan, X.-H. and Nguyen, C.-T. (2006). Jgibblda: A java implementation of latent dirichlet allocation (lda) using gibbs sampling for parameter estimation and inference. Retrieved from http://jgibblda.sourceforge.net.

Reicher, S. D., Spears, R., and Postmes, T. (1995). A social identity model of deindividuation phenomena. *European review of social psychology*, 6(1):161–198.

Spears, R. and Lea, M. (1992). *Social influence and the influence of the'social'in computer-mediated communication.* Harvester Wheatsheaf.

Spears, R. and Lea, M. (1994). Panacea or panopticon? the hidden power in computer-mediated communication. *Communication Research*, 21(4):427–459.

Ting, I.-H., Chi, H.-M., Wu, J.-S., and Wang, S.-L. (2013). An approach for hate groups detection in facebook. In *The 3rd International Workshop on Intelligent Data Analysis and Management*, pages 101–106. Springer.

Tokunaga, R. S. (2010). Following you home from school: A critical review and synthesis of research on cyberbullying victimization. *Computers in human behavior*, 26(3):277–287.

Warner, W. and Hirschberg, J. (2012). Detecting hate speech on the world wide web. In *Proceedings of the Second Workshop on Language in Social Media*, pages 19–26. Association for Computational Linguistics.

Xiang, G., Fan, B., Wang, L., Hong, J., and Rose, C. (2012). Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1980–1984. ACM.

A Dictionary-based Approach to Racism Detection in Dutch Social Media

Stéphan Tulkens, Lisa Hilte, Elise Lodewyckx, Ben Verhoeven, Walter Daelemans
CLiPS Research Center, University of Antwerp
Prinsstraat 13, 2000, Antwerpen, Belgium
{stephan.tulkens, lisa.hilte, ben.verhoeven, walter.daelemans}@uantwerpen.be,
elise.lodewyckx@student.uantwerpen.be

Abstract

We present a dictionary-based approach to racism detection in Dutch social media comments, which were retrieved from two public Belgian social media sites likely to attract racist reactions. These comments were labeled as racist or non-racist by multiple annotators. For our approach, three discourse dictionaries were created: first, we created a dictionary by retrieving possibly racist and more neutral terms from the training data, and then augmenting these with more general words to remove some bias. A second dictionary was created through automatic expansion using a `word2vec` model trained on a large corpus of general Dutch text. Finally, a third dictionary was created by manually filtering out incorrect expansions. We trained multiple Support Vector Machines, using the distribution of words over the different categories in the dictionaries as features. The best-performing model used the manually cleaned dictionary and obtained an F-score of 0.46 for the racist class on a test set consisting of unseen Dutch comments, retrieved from the same sites used for the training set. The automated expansion of the dictionary only slightly boosted the model's performance, and this increase in performance was not statistically significant. The fact that the coverage of the expanded dictionaries did increase indicates that the words that were automatically added did occur in the corpus, but were not able to meaningfully impact performance. The dictionaries, code, and the procedure for requesting the corpus are available at: `https://github.com/clips/hades`.

Keywords: Racism, word2vec, Dictionary-based Approaches, Computational Stylometry

1. Introduction

Racism is an important issue which is not easily defined, as racist ideas can be expressed in a variety of ways. Furthermore, there is no clear definition of what exactly constitutes a racist utterance; what is racist to one person is highly likely to not be considered racist universally. Additionally, although there exist mechanisms for reporting acts of racism, victims often neglect to do so as they feel that reporting the situation will not solve anything, according to the Belgian Interfederal Centre for Equal Opportunities (CNTR).[1] The scope of this issue, however, is currently unknown. Hence, the goal of our system is two-fold: it can be used to shed light on how many racist remarks are not being reported online, and furthermore, the automated detection of racism could provide interesting insights in the linguistic mechanisms used in racist discourse.

In this study, we try to automatically detect racist language in Dutch social media comments, using a dictionary-based approach. We retrieved and annotated comments from two public social media sites which were likely to attract racist reactions according to the CNTR. We use a Support Vector Machine to automatically classify comments, using handcrafted dictionaries, which were later expanded using automated techniques, as features.

We first discuss previous research on our subject and methodology, and discuss the problem of defining racist language (section 2). Next, we describe our data (section 3). Finally, after discussing the experimental setup (section 4), we present our results (section 5).

2. Related Research

The classification of racist insults presents us with the problem of giving an adequate definition of racism. More so than in other domains, judging whether an utterance is an act of racism is highly personal and does not easily fit a simple definition. The Belgian anti-racist law forbids discrimination, violence and crime based on physical qualities (like skin color), nationality or ethnicity, but does not mention textual insults based on these qualities.[2] Hence, this definition is not adequate for our purposes, since it does not include the racist utterances one would find on social media; few utterances that people might perceive as racist are actually punishable by law, as only utterances which explicitly encourage the use of violence are illegal. For this reason, we use a *common sense* definition of racist language, including all negative utterances, negative generalizations and insults concerning ethnicity, nationality, religion and culture. In this, we follow Orrù (2015), Bonilla-Silva (2002) and Razavi et al. (2010), who show that racism is no longer strictly limited to physical or ethnic qualities, but can also include social and cultural aspects.

Additionally, several authors report linguistic markers of racist discourse; Van Dijk (2002) reports that the number of available topics is greatly restricted when talking about foreigners. Orrù (2015), who performed a qualitative study of posts from Italian social media sites, shows that these chosen topics are typically related to migration, crime and economy. Furthermore, the use of stereotypes and prejudiced statements (Reisigl and Wodak, 2005; Quasthoff, 1989), as well as a heightened occurrence of truth claims (Greevy and Smeaton, 2004b; Greevy and Smeaton, 2004a), are reported as typical characteristics of racist discourse . Finally, racist utterances are said to contain specific words and phrases, i.e. n-grams, significantly more often than neutral texts, like "our own kind" and

[1] `http://www.diversiteit.be`

[2] `http://www.diversiteit.be/de-antiracismewet-van-30-juli-1981`

"white civilization" (Greevy and Smeaton, 2004b; Greevy and Smeaton, 2004a).

Stylistically, racist discourse is characterized by a higher rate of certain word classes, like imperatives and adjectives and a higher noun-adjective ratio (Orrù, 2015; Greevy and Smeaton, 2004b; Greevy and Smeaton, 2004a). Greevy and Smeaton also report a more frequent use of modals and adverbs, which they link to the higher frequency of truth claims in racist utterances (2004b; 2004a). In several studies, pronoun use is reported as an important feature in the detection of racist language. While Orrù (2015) reports a high frequency of (especially first person plural) pronouns in racist data, Van Dijk (2002) reports a more general finding: the importance of *us and them* constructions in racist discourse. He explains that they involve a 'semantic move with a positive part about Us and a negative part about Them' (Van Dijk, 2002, p.150). Using such constructions, one linguistically emphasizes - either deliberately or subconsciously - a divide between groups of people. A strict interpretation implies that even positive utterances about 'them' can be perceived as racist, as they can also imply a divide between us and them. In this sense, Van Dijk's definition of racism is subtler, but also broader, than the definition used in our own research: we only count *negative* utterances and generalizations about groups of people as racist.

Our dictionary-based approach is inspired by methods used in previous research, like LIWC (Linguistic Inquiry and Word Count) (Pennebaker et al., 2001). LIWC is a dictionary-based computational tool that counts word frequencies for both grammatical categories (e.g. pronouns) and content-related categories (e.g. negative emotion words). As LIWC uses counts per category instead of individual words' frequencies, it allows for broader generalizations on functionally or semantically related words.

The construction of dictionary categories related to racist discourse (cf. section 4.1) is largely based on linguistic properties of racist language reported in earlier work (see above). Additionally, the categories were adjusted to fit the corpus used in the research, which differs from corpora used in other studies. As our corpus is retrieved from social media sites with an anti-Islamic orientation, we added categories to reflect anti-religious sentiment. The relevant features in this study therefore differ from those reported in other studies, as different words are used to insult different groups of people (Greevy and Smeaton, 2004a).

Finally, some other successful quantitative approaches to racism detection that have been used in earlier studies are a bag of words (BoW) approach as well as the analysis of part-of-speech (PoS) tags (Greevy and Smeaton, 2004b; Greevy and Smeaton, 2004a). We leave the addition of these features to future work.

3. Datasets and Annotations

In this section, we describe our data collection, our annotation guidelines (3.1) and the results of our annotations (3.2 and 3.3).

For our current research we collected a corpus of social media comments, consisting of comments retrieved from Facebook sites which were likely to attract racist reactions in their comments. We specifically targeted two sites: the site of a prominent Belgian anti-Islamic organization, and the site of a Belgian right-wing organization. In both cases the Facebook sites were officially condoned by the organizations, and in the first case served as a communication platform to organize political gatherings. While both sites, the former more than the latter, explicitly profess to be non-racist, the comments they attracted were still highly critical of foreigners and, predictably, Muslims. This is also the reason we mined *comments* from these sites, and not the posts themselves. While the narrow focus of the sites introduces bias into our data, as the opinions of the people visiting these sites will not reflect the opinions of the general population, they do contain a good proportion of racist to non-racist data.

3.1 Annotation Style

We annotated the retrieved comments with three different labels: 'racist', 'non-racist' and 'invalid'.

The 'racist' label describes comments that contain negative utterances or insults about someone's ethnicity, nationality, religion or culture. This definition also includes utterances which equate, for example, an ethnic group to an extremist group, as well as extreme generalizations. The following examples are comments that were classified as racist:

1. Het zijn precies de vreemden die de haat of het racisme opwekken bij de autochtonen.
   It is the foreigners that elicit hate and racism from natives.

2. Kan je niets aan doen dat je behoort tot het ras dat nog minder verstand en gevoelens heeft in uw hersenen dan het stinkend gat van een VARKEN ! :-p
   You cannot help the fact that you belong to the race that has less intellect and sense in their brains than the smelly behind of a PIG! :-P

3. Wil weer eens lukken dat wij met het vuilste krapuul zitten, ik verschiet er zelfs niet van!
   Once again we have to put up with the filthiest scum, it doesn't even surprise me anymore!

The label 'invalid' was used for comments that were written in languages other than Dutch, or that did not contain any textual information, i.e. comments that solely consist of pictures or links. Before classification, we excluded these from both our training and test set.

The final label, 'non-racist', was the default label. If a comment was valid, but could not be considered racist according to our definition, this was the label we used.

3.2 Training Data

To collect the training data, we used `Pattern`[3] (De Smedt and Daelemans, 2012) to scrape the 100 most recent posts from both sites, and then extracted all comments which reacted to these comments. This resulted in 5759 extracted comments: 4880 from the first site and 879 from the second site. The second site attracted a lot less comments on each post, possibly because the site posted more frequently.

[3] `http://www.clips.uantwerpen.be/pattern`

	# Train Comments	# Test Comments
Non-racist	4500	443
Racist	924	164
Invalid	335	9

Table 1: Gold standard corpus sizes.

In addition to this, the organization behind the first site had been figuring prominently in the news at the time of extraction, which might explain the divide in frequency of comments between the two sites. The corpus was annotated by two annotators, who were both students of comparable age and background. When A and B did not agree on a label, a third annotator, C, was used as a tiebreaker in order to obtain gold-standard labels. Table 1 shows the gold standard for the training set.

We calculated inter-annotator agreement using the Kappa score (κ) (Cohen, 1968). On the training corpus, the agreement score was $\kappa = 0.60$. Annotator A used the racist tag much less often than annotator B. Interestingly, the agreement remains relatively high; 79% of the comments that A annotated as racist were also annotated as racist by B. Even though B was much more inclined to call utterances racist, A and B still shared a common ground regarding their definition of racism. Examining the comments in detail, we found that the difference can largely be explained by sensitivity to insults and generalizations, as example 4 shows.

4. Oprotten die luizegaards [*sic*] !!!
   Throw those lice carriers out!

While annotator B considers this utterance to be racist, annotator A does not, as it does not contain a specific reference to an ethnicity, nationality or religion. That is, when not seen in the context of this specific annotation task this sentence would not necessarily be called racist, just insulting.

3.3 Test data

The test corpus was mined in the same way as the training set, at a different point in time. We mined the first 500 and first 116 comments from the first and second site, respectively, which makes the proportion between sites more or less identical to the the proportions in the train corpus. The annotation scheme was identical to the one for the train set, with the difference that C, who previously performed the tiebreak, now became a regular annotator. The first 25% of each batch of comments, i.e. 125 comments for the first site and 30 comments for the second site, were annotated by all three annotators to compute inter-annotator agreement. The remaining comments were equally divided among annotators. The annotator agreement was $\kappa = 0.54$ (pairwise average), which is lower than the agreement on the training data. The reason for the lower agreement was that annotator C often did not agree with A and B. Because the pattern of mismatches between the annotators is quite regular, we will now discuss some of the annotations in detail:

5. we kunnen niet iedereen hier binnen laten want dat betekend [*sic*] het einde van de europese beschaving
   We cannot let everyone in because that will mean the end of European civilization

6. Eigen volk gaat voor, want die vuile manieren van de EU moeten wij vanaf. Geen EU en geen VN. Waardeloos en tegen onze mensen. (eigen volk.)
   Put our own people first, because we need to get rid of the foul manners of the EU. No EU nor UN. Useless and against our people. (own folk.)

7. Burgemeester Termont is voor de zwartzakken die kiezen voor hem
   Mayor Termont supports the black sacks, as they vote for him

Annotator C used the 'racist' tag more often, which is probably due to the fact that he consistently annotated overt ideological statements related to immigration as 'racist', while the other annotators did not. The three examples mentioned above are utterances that C classified as 'racist', but A and B classified as 'not racist'.

The cause of these consistent differences in annotations might be cultural, as C is from the southern part of the Netherlands, whereas A and B are native to the northern part of Belgium. Some terms are simply misannotated by C because they are Flemish vernacular expressions. For example, `zwartzak` [black sack], from sentence 7, superficially looks like a derogatory term for a person of color, but actually does not carry this meaning, as it is a slang word for someone who collaborated with the German occupying forces in the Second World War. While this could still be classified as being racist, the point is that C only registered this as a slang word based on skin color, and not a cultural or political term. Finally, it is improbable that the cause of these mismatches is annotator training, as A and B did not discuss their annotations during the task. In addition to this, C functioned as a tiebreaker in the first dataset, and thus already had experience with the nature of the training material.

4. Experimental Setup

In this section, we describe our experimental setup. We will first discuss our dictionary-based approach, describing both the LIWC dictionary we used as well as the construction of dictionaries related to racist discourse (section 4.1). Next, we will describe the preprocessing of the data (section 4.2).

4.1 Dictionaries

4.1.1 LIWC

In our classification task, we will use the LIWC dictionaries for Dutch[4] (Zijlstra et al., 2004). We hypothesize that some of LIWC's word categories can be useful in detecting (implicit) racist discourse, as some of these categories are associated with markers of racist discourse reported in previous research (cf. section 2), including pronouns, negative emotion words, references to others, certainty, religion and curse words.

[4] An exhaustive overview of all categories in the Dutch version of LIWC can be found in Zijlstra et al. (2004, p. 277-278).

	Negative	Neutral
Skin color	✓	✓
Nationality	✓	✓
Religion	✓	✓
Migration	✓	✓
Country	✓	✓
Stereotypes	✓	
Culture	✓	
Crime	✓	
Race	✓	
Disease	✓	

Table 2: Overview of the categories in the discourse dictionary

4.1.2 Discourse Dictionaries

In addition to the Dutch LIWC data, we created a dictionary containing words that specifically relate to racist discourse. We expect a dictionary-based approach in which words are grouped into categories to work well in this case because many of the racist terms used in our corpus were neologisms and hapaxes, like `halalhoer` (halal prostitute). Alternatively, existing terms are often reused in a ridiculing fashion, e.g. using the word `mossel` (mussel) to refer to Muslims. The dictionary was created as follows: after annotation, terms pertaining to racist discourse were manually extracted from the training data. These were then grouped into different categories, where most categories have both a neutral and a negative subcategory. The negative subcategory contains explicit insults, while the neutral subcategory contains words that are normally used in a neutral fashion, e.g. `zwart` (black), `Marokkaan` (Moroccan), but which might also be used in a more implicit racist discourse; e.g. people that often talk about nationalities or skin color might be participating in a racist us and them discourse. An overview of the categories can be found in Table 2.

After creating the dictionary, we expanded these word lists both manually and automatically. First, we manually added an extensive list of countries, nationalities and languages, to remove some of the bias present in our training corpus. To combat sparsity, and to catch productive compounds which are likely to be used in a racist manner, we added wildcards to the beginning or end of certain words. We used two different wildcards. `*` is an inclusive wildcard; it matches the word with or without any affixes, e.g. `moslim*` matches both `moslim` (Muslim) and `moslims` (Muslims). `+` is an exclusive wildcard; it only matches words when an affix is attached, e.g. `+moslim` will match `rotmoslim` (Rotten Muslim) but not `moslim` by itself. In our corpus (which is skewed towards racism), the + will almost always represent a derogatory prefix, which is why it figures more prominently in the negative part of our dictionary.

A downside of using dictionaries for the detection of racism, is that they do not include a measure of context. Therefore, a sentence such as "My brother hated the North African brown rice and lentils we made for dinner"[5] will be classified as racist, regardless of the fact that the words above do not occur in a racist context. Approaches based on word unigrams or bigrams face similar problems. This problem is currently partially absolved by the fact that we are working with a corpus skewed towards racism: words like 'brown' and 'African' are more likely to be racist words in our corpus than in general text.

[5]We thank an anonymous reviewer for suggesting the sentence.

	#Words
Original	1055
Expanded	3845
Cleaned	3532

Table 3: Dictionary word frequencies.

4.1.3 Automated Dictionary Expansion

To broaden the coverage of the categories in our dictionary, we performed dictionary expansion on both the neutral and the negative categories using `word2vec` (Mikolov et al., 2013). `word2vec` is a collection of models capable of capturing semantic similarity between words based on the sentential contexts in which these words occur. It does so by projecting words into an n-dimensional space, and giving words with similar contexts similar places in this space. Hence, words which are closer to each other as measured by cosine distance, are more similar. Because we observed considerable semantic variation in the insults in our corpus, we expect that dictionary expansion using `word2vec` will lead to the extraction of previously unknown insults, as we assume that similar insults are used in similar contexts. In parallel, we know that a lot of words belonging to certain semantic categories, such as diseases and animals, can almost invariably be used as insults.

The expansion proceeded as follows: for each word in the dictionary, we retrieved the five closest words, i.e. the five most similar words, in the n-dimensional space, and added these to the dictionary. Wildcards were not taken into account for this task, e.g. `*jood` was replaced by `jood` for the purposes of expansion. As such, the expanded words do not have any wildcards attached to them. For expansion we used the best-performing model from Tulkens et al. (2016), which is based on a corpus of 3.9 billion words of general Dutch text. Because this `word2vec` model was trained on general text, the semantic relations contained therein are not based on racist or insulting text, which will improve the coverage of our expanded categories.

After expansion, we manually searched the expanded dictionaries and removed obviously incorrect items. Because the `word2vec` model also includes some non-Dutch text, e.g. Spanish, some categories were expanded incorrectly. As a result, we have 3 different dictionaries with which we perform our experiments: the original dictionary which was based on the training data, a version which was expanded using `word2vec`, and a cleaned version of this expanded version. The word frequencies of the dictionaries are given in Table 3. An example of expansion is given in Table 4.

4.2 Preprocessing and Featurization

For preprocessing, the text was first tokenized using the Dutch tokenizer from `Pattern` (De Smedt and Daele-

Dictionary	Example
Original:	mohammed*
Expanded:	mohammed*, **mohamed, mohammad, muhammed, vzmh, hassan**
Cleaned:	mohammed*, mohamed, mohammad, muhammed, ~~vzmh~~, hassan

Table 4: An example of expansion. The original dictionary only contains a single word. In the expanded version, the **bold** words have been added. In the third version the words that were ~~struck through~~ have been removed.

mans, 2012), and then lowercased and split on whitespace, which resulted in lists of words which are appropriate for lexical processing.

Our dictionary-based approach, like LIWC, creates an *n*-dimensional vector of normalized and scaled numbers, where *n* is the number of dictionary categories. These numbers are obtained by dividing the frequency of words in every specific category by the total number of words in the comment. Because all features are already normalized and scaled, there was no need for further scaling. Furthermore, because the number of features is so small, we did not perform explicit feature selection.

5. Results and Discussion

5.1 Performance on the Training Set

We estimated the optimal values for the SVM parameters by an exhaustive search through the parameter space, which led to the selection of an RBF kernel with a C value of 1 and a gamma of 0. For the SVM and other experiments, we used the implementation from `Scikit-Learn` (Pedregosa et al., 2011). Using cross-validation on the training data, all dictionary-based approaches with lexical categories related to racist discourse significantly outperformed models using only LIWC's general word categories. Since the current research concerns the binary classification of racist utterances, we only report scores for the positive class, i.e. the racist class. When only LIWC-categories were used as features, an F-score of 0.34 (std. dev. 0.07) was obtained for the racist class. When using the original discourse dictionary, we reached an F-score of 0.50 (std. dev. 0.05). Automatic expansion of the categories did not influence performance either (F-score 0.50, std. dev. 0.05). Similar results (0.49 F-score, std. dev. 0.05) were obtained when the expanded racism dictionaries were manually filtered. This result is not surprising, as the original dictionaries were created from the training data, and might form an exhaustive catalog of racist terms in the original corpus.

Combining the features generated by LIWC with the specific dictionary-based features led to worse results compared to the dictionary-based features by themselves (F-score 0.40, std. dev. 0.07 for the best-performing model).

Finally, all models based on the dictionary features as well as the combined model outperformed a unigram baseline of 0.36, but the LIWC model did not. We also report a weighted random baseline (WRB), which was outperformed by all models.

	P	R	F
Original	**0.42**	0.61	**0.50**
Expanded	0.40	**0.64**	**0.50**
Cleaned	0.40	**0.64**	0.49
LIWC	0.27	0.47	0.34
Combined	0.36	0.44	0.40
Unigram	0.38	0.34	0.36
WRB	0.27	0.27	0.27

Table 5: Results on the train set. WRB is a weighted random baseline.

5.2 Testing the Effect of Expansion

As seen above, the performance of the different models on the train set was comparable, regardless of their expansion. This is due to the creation procedure for the dictionary: because the words in the original dictionary were directly retrieved from the training data, the expanded and cleaned versions might not be able to demonstrate their generalization performance, as most of the racist words from the training data will be included in the original dictionaries as well as the expanded dictionaries. This artifact might disappear in the test set, which was retrieved from the same two sites, but will most likely contain unseen words. These unseen words will not be present in the original dictionary, but could be present in the expanded version.

As Table 6 shows, the models obtain largely comparable performance on the test set, and outperform the unigram baseline by a wide margin. In comparison to previous research, our approach leads to worse results than those of Greevy and Smeaton (2004a), who report a precision score of 0.93 and a recall score of 0.87, using an SVM with BOW features together with frequency-based term weights. It is, however, difficult to compare these scores to our performance, given that the data, method, and language differ.

Our best-performing model was based on the expanded and cleaned version of the dictionary, but this model only slightly outperformed the other models. Additionally, we also computed Area Under the Receiving Operator Characteristic Curve (ROC-AUC) scores for all models, also shown in Table 6. ROC-AUC shows the probability of ranking a randomly chosen positive instance above a randomly chosen negative instance, thereby giving an indication of the overall performance of the models. This shows that all dictionaries have comparable AUC scores, and that each dictionary outperforms the unigram baseline. To obtain additional evidence, we computed the statistical significance of performance differences between the models based on the dictionaries and unigram baseline model using approximate randomization testing (ART) (Noreen, 1989).[6] An ART test between dictionary models reveals that none of the models had performance differences that were statistically significant. Similarly, all dictionary models outperformed the unigram baseline with statistical significance, with $p < 0.01$ for the models based on the cleaned and expanded dictionaries, and $p < 0.05$ for the models based on

[6] We used the implementation by Vincent Van Asch, which is available from the CLiPS website `http://www.clips.uantwerpen.be/scripts/art`

	P	R	F	AUC
Original	**0.51**	0.39	0.44	**0.63**
Expanded	0.48	**0.43**	0.45	**0.63**
Cleaned	0.49	**0.43**	**0.46**	**0.63**
Unigram	0.46	0.20	0.28	0.56

Table 6: P, R, F and ROC-AUC scores on the test set.

	% Coverage	# comments	# racist
Original	0.014	98	43
Expanded	0.035	212	82
Cleaned	0.034	206	81

Table 7: Coverage of the various dictionaries in vocabulary percentage, number of comments, and number of racist comments.

the original dictionary.

To get more insight into why the expanded models were not more successful, we calculated dictionary coverage for every dictionary separately on the test set. If the expanded dictionaries do not have increased coverage, the reason for their similar performance is clear: not enough words have been added to affect the performance in any reasonable way. As Table 7 indicates, the coverage of the expanded dictionaries did increase, which indicates that the automated expansion, or manual deletion for that matter, contrary to expectations, did not add words that were useful for the classification of racist content. To obtain additional evidence for this claim, we looked at the number of comments that contained words from the original, cleaned and expanded dictionaries. The coverage in terms of total comments also increased, as well as the absolute number of racist comments that contained the added terms. Because the coverage in number of comments did not increase the performance of the dictionaries, we hypothesize that the terms that were included in the expanded dictionaries were not distributed clearly enough (over racist and neutral texts) to make a difference in the performance on the classification task.

6. Conclusions and Future Work

We developed a dictionary-based computational tool for automatic racism detection in Dutch social media comments. These comments were retrieved from public social media sites with an anti-Islamic orientation. The definition of racism we used to annotate the comments therefore includes religious and cultural racism as well, a phenomenon reported on in different studies (Orrù, 2015; Bonilla-Silva, 2002; Razavi et al., 2010).

We use a Support Vector Machine to classify comments as racist or not based on the distribution of the comments' words over different word categories related to racist discourse. To evaluate the performance, we used our own annotations as gold standard. The best-performing model obtained an F-score of 0.46 for the racist class on the test set, which is an acceptable decrease in performance compared to cross-validation experiments on the training data (F-score 0.49, std. dev. 0.05). The dictionary used by the model was manually created by retrieving possibly racist and more neutral terms from the training data during annotation. The dictionary was then manually expanded, automatically expanded with a `word2vec` model and finally manually cleaned, i.e. irrelevant terms that were added automatically were removed. It did not prove useful to use general stylistic or content-based word categories along with the word lists specifically related to racist discourse.

Surprisingly, the expansion of the manually crafted dictionary did not boost the model's performance significantly. In (cross-validated) experiments on the training data, this makes sense, as the words in the different categories are retrieved from the training data itself, artificially making the dictionary very appropriate for the task. In the test runs, however, a better result could be expected from the generalized word lists. The expanded versions of the dictionary had higher overall coverage for the words in the corpus, as well as higher coverage in number of comments and in number of racist comments. This shows that the words that were automatically added, did indeed occur in our corpus. As the model's performance more or less stagnated when using the expanded categories compared to the original ones, we hypothesize that the terms that were automatically added by the `word2vec` model were irrelevant to the task of discriminating between racist and neutral texts.

In terms of future work, we will expand our research efforts to include more general social media text. Because we currently only use material which was gathered from sites skewed towards racism, the performance of our dictionary might have been artificially heightened, as the words in the dictionary only occur in racist contexts in our corpus. Therefore, including more general social media texts will serve as a good test of the generality of our dictionaries with regards to detecting insulting material.

7. Acknowledgments

We are very grateful towards Leona Erens and François Deleu from the Interfederal Centre for Equal Opportunities (CNTR) for wanting to collaborate with us and for pointing us towards the necessary data. We thank the three anonymous reviewers for their helpful comments and advice.

8. Supplementary Materials

The supplementary materials are available at `https://github.com/clips/hades`

9. Bibliographical References

Bonilla-Silva, E. (2002). The linguistics of color blind racism: How to talk nasty about blacks without sounding "racist". *Critical Sociology*, 28(1-2):41–64.

Cohen, J. (1968). Weighted Kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin*, 70(4):213.

De Smedt, T. and Daelemans, W. (2012). Pattern for Python. *The Journal of Machine Learning Research*, 13(1):2063–2067.

Greevy, E. and Smeaton, S. (2004a). Text categorization of racist texts using a support vector machine. *7 es Journées internationales d'Analyse statistique des Données Textuelles*.

Greevy, E. and Smeaton, A. F. (2004b). Classifying racist texts using a support vector machine. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 468–469. ACM.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Noreen, E. (1989). Computer-intensive methods for testing hypotheses: an introduction.

Orrù, P. (2015). Racist discourse on social networks: A discourse analysis of Facebook posts in Italy. *Rhesis*, 5(1):113–133.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pennebaker, J. W., Francis, M. E., and Booth, R. J. (2001). Linguistic inquiry and word count: LIWC 2001. *Mahway: Lawrence Erlbaum Associates*, 71:2001.

Quasthoff, U. (1989). Social prejudice as a resource of power: Towards the functional ambivalence of stereotypes. *Wodak, R.(éd.), Language, Power and Ideology. Amsterdam: Benjamins*, pages 181–196.

Razavi, A. H., Inkpen, D., Uritsky, S., and Matwin, S. (2010). Offensive language detection using multi-level classification. In *Advances in Artificial Intelligence*, pages 16–27. Springer.

Reisigl, M. and Wodak, R. (2005). *Discourse and discrimination: Rhetorics of racism and antisemitism*. Routledge.

Tulkens, S., Emmery, C., and Daelemans, W. (2016). Evaluating unsupervised Dutch word embeddings as a linguistic resource. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC)*. European Language Resources Association (ELRA).

Van Dijk, T. A. (2002). Discourse and racism. *The Blackwell companion to racial and ethnic studies*, pages 145–159.

Zijlstra, H., Van Meerveld, T., Van Middendorp, H., Pennebaker, J. W., and Geenen, R. (2004). De Nederlandse versie van de 'linguistic inquiry and word count'(LIWC). *Gedrag & gezondheid*, 32:271–281.

Forensic Investigation of Linguistic Sources of Electronic Scam Mail: A Statistical Language Modelling Approach

Adeola O Opesade[1], **Mutawakilu A Tiamiyu**[1], **Tunde Adegbola**[2]
[1] Africa Regional Centre for Information Science, University of Ibadan, Nigeria
[2] African Languages Technology-Initiative (ALT-I), Ibadan, Nigeria
E-mail: morecrown@gmail.com, mutatiamiyu@yahoo.com, taintransit@hotmail.com

Abstract

Electronic handling of information is one of the defining technologies of the digital age. These same technologies have been exploited by unethical hands in what is now known as cybercrime. Cybercrime is of different types but of importance to the present study is the 419 Scam because it is generally (yet controversially) linked with a particular country - Nigeria. Previous research that attempted to unravel the controversy applied the Internet Protocol address tracing technique. The present study applied the statistical language modelling technique to investigate the propensity of Nigeria's involvement in authoring these fraudulent mails. Using a hierarchical modelling approach proposed in the study, 28.85% of anonymous electronic scam mails were classified as being from Nigeria among four other countries. The study concluded that linguistic cues have potentials of being used for investigating transnational digital breaches and that electronic scam mail problem cannot be pinned down to Nigeria as believed generally, though Nigeria could be one of the countries that are prominent in authoring such mails.

Keywords: digital forensics, 419 scam, statistical language modelling

1. Introduction

1.1 Background to the Study

Electronic handling of information is one of the defining technologies of the digital age. The age is characterized by the application of computer technology as a transformative tool to enhance effectiveness and efficiency in personal, commercial, educational, governmental, and other facets of modern life (Reith, Carr, and Gunsch, 2002). However, alongside the benefits derivable from the ability to automatically manage so much information are threats to communications and transactions conducted electronically. Just as ICT provides new opportunities to operate and expand one's presence and reach, it also presents opportunities for those with criminal intentions, leaving its numerous users highly exposed to the threat of cyber attack and cybercrime (Choo, 2011).

The term cybercrime encompasses traditional crimes such as fraud, scam, theft, forgery, harassment, blackmail, embezzlement, in as much as the computer or its network is employed for perpetration; it also involves a host of new criminal activities which cannot be perpetrated but with computer or its network, such as spam, denial of service attacks and distribution of viruses (Torosyan, 2003; Alshalan, 2006).

The advance fee fraud (also known as Internet Scam, Nigerian 419 Scam or *Yahoo-Yahoo*) is a type of internet fraud that is generally believed to have originated from Nigeria and perpetrated mostly by Nigerians. This stance has however, been faced with much controversy. Perceptions on the extent of Nigeria's involvement in the current status of advance fee fraud can be categorised into three; while the first two views are distinctly opposite, the third stands somewhere in between them. Holding the first view are sources that believe that Nigeria is the main, if not the sole source, of this fraud (State Department Publication, 1997). Holding the second view are sources that refute the fact that Nigeria is as prominent in cybercrime as portrayed by those other reports. They argued that the increased consciousness about crime and the 419 scams in particular is only a motive to criminalize the Nigerian state (Bayart, Ellis and Hibou, 1999). Holding the third view are those of the opinion that the term "Nigerian Advance Fee Fraud" is only partially accurate and the problem is truly one of international dimension with victims and offenders being located across the globe (Smith, Holmes and Kaufmann, 2001).

Previous studies that focused on unravelling the controversy used e-mail header (Edelson, 2003) and Internet Protocol (IP) address tracing technique (Longe and Osofisan, 2011). Effective as their methods are however, they are constrained by geographical boundaries and cannot detect the nationality of a scammer who sends an electronic scam mail from outside the shores of his own country. The methods are also not effective against some computer security frauds such as phishing, spoofing and masquerading. There is therefore, a need for new methodologies for investigating sources of such fraudulent e-mail.

Dyrud (2005) analysed ninety-three (93) electronic scam letters (which he called Nigerian scam letters) which he received over a period of 10 months. While analysing one of the letters he commented:

> *Some are ludicrously transparent, such as John Kredoski's November, November 13, 2004 e-mail. He lives in Reno, has suffered a heart attack, and has chosen to write to me because I am "a fellow American." "Do not," he cautions, "associates this letter as numerous letters we receive from Africa. If you need my passport to prove my identity, I will send it to you." For a native-born American, his syntax and vocabulary are curiously similar to his Nigerian counterparts.* (Dyrud, 2005: p. 7).

Although Dyrud (2005) did not present any objective means of deducing that the received mails were Nigerian, he was able to differentiate the language usage of the writer from his own American style, despite the writer's claim of being an American. Exploitation of linguistic cues can therefore, aid in further detection of sources of anonymous

electronic mails; this is because a writer's natural language is less susceptible to deception.

Language is an intricate system of structural components for encoding and decoding information (Cruz-Ferreira and Abraham, 2006). All human languages exhibit a great deal of internal variations in terms of a specific set of linguistic items or human speech patterns such as words, idiosyncratic, structural or grammatical features which can uniquely associate with some external factors such as geographical area or social group, time, situation, genre, subject, author amongst others (Adetugbo, 1979; Banjo, 1979; Wardhaugh, 1994). The phenomenon of Standard Nigerian Spoken English was proposed by Banjo in 1971 (Jowiit, 1991). According to Adetugbo (1979) the fact that English language in Nigeria is regarded as a dialect of English means that it preserves the same common core items of vocabulary and structure found in other varieties of English everywhere else, however, the implications of spatial dialect differentiation, the mode of acquisition of the English language in Nigeria, interference of native languages on English in its second language situation in Nigeria and the non-equivalence of the social and cultural environment in Nigeria with that of any other English language speaking community, demanded that communicative competence in English in Nigeria cannot be the same as for any other variety of English.

The features of the Nigerian variant of the language have been well researched in the humanities, for example, on the level of morphology, due to the influence of the dominant local language or mother tongue the spoken English shows variations in accent. On the lexical level, the problem with the use of the article (definite and indefinite) is endemic (Egbe, 1979). Kujore (1985) and Jowiit (1991) composed specific variations alongside the standard British English in order to show the growing divergence between the Nigerian usage and British usage of the language. Opesade, Adegbola and Tiamiyu (2013) validated quantitatively the presence of English language variants of five African countries, Cameroon, Ghana, Liberia, Nigeria and Sierra-Leone. If Nigerian and other countries' variants of the English language exist, then this could be a means of identifying Nigerian texts written in the English language from non-Nigerian text written in the other variants of same language.

1.2 Objective of the Study

The objective of the study was to establish and apply statistical language models for the detection of electronic scam mail likely to be of Nigerian origin, based on linguistic cues. To achieve the objective of the study, the following sub-objectives were pursued:

1. To determine a precise model for separating electronic texts of Nigerian linguistic origin from those of other countries (Cameroon, Ghana, Liberia, Sierra-Leone).
2. 2. To apply the model to classify electronic scam mail into countries of origin based on linguistic cues.

1.3 Research Questions

1. What is the performance of the (NigGh_Other) model that classifies Nigerian and Ghanaian texts from those from the other three countries (Liberia, Cameroon and Sierra Leone)?
2. What is the performance of the (Nig_Gh) model that classifies Nigerian and Ghanaian texts only?
3. What percentage of electronic scam mail are classified to be Nigerian based on linguistic cues.

3. Research Methodology

3.1 Research Design

The research was carried out using modelling and simulation methodologies. In this study, online English texts of five African countries were modelled statistically. Out of a population of all English speaking countries in West Africa including Cameroon, the domains of estimation selected for this study were Nigeria, Ghana, Liberia, Cameroon and Sierra-Leone. Nigeria is the country of focus in this investigation, while Ghana, Liberia, Cameroon and Sierra-Leone are selected for the purpose of comparing their English language usages to that of Nigeria as a means of providing alternative candidate countries in the attribution of the electronic scam mails.

3.2 Target Populations

To achieve the objective of this study, two populations were examined. They were:

i. All readers' comments posted on the fifty–two (52) pages contained (as of November 2011) in *www.topix.com* websites of the five countries selected for the study and the additional four pages posted (as of February 2012) in the same websites.
ii. All eight hundred and seventy-three (873) electronic scam mails available in scam archives such as scam bating sites as of 14th March 2011. Scam baiting, also known as counter scamming, is the practice of feigning interest in a fraudulent scheme in order to manipulate a scammer.

3.3 Sampling Techniques for Composition of Study Corpora

A multistage sampling technique was employed for textual data collection. The sampling procedure for each of the two populations is as presented below:

3.3.1 Sampling Technique for the www.topix.com Texts

A multistage sampling technique was used to select a representative sample of electronic texts from the population of texts contained in the relevant country pages of the website *www.topix.com*. To get the texts that could be useful for a supervised learning approach of the study, each text was opened, read and assessed based on the number of words contained and a sense of affiliation to the respective country as depicted in the content. A comment was considered to be affiliated to (and labelled to be from) a particular country if it was found in that country's forum and if it contained such phrases as 'our country', 'our beloved country' and other related ones in its discourse. Initially the researchers targeted selecting texts with a hundred or more words; however, this was reduced to texts with twenty (20) or more words because of the scarcity of large texts on the discussion forums. The numbers of texts selected for the study in November 2011 and based on the assessment criteria are as shown in Table 1.

Another set of one hundred and fifteen (115) texts were collected from the country forums of *www.topix.com* in February 2012. A complete enumeration of all texts on the

Country's forum website	No. of pages	Pages selected	No. of selected texts
www.topix.com/forum/world/nigeria	31	2,8,13,25	425
www.topix.com/forum/world/ghana	9	2,3,6.9	317
www.topix.com/forum/world/liberia	4	1-4	130
www.topix.com/forum/world/cameroon	4	1-4	241
www.topix.com/forum/world/sierra-leone	4	1-4	357
Total no. of Texts			1,470

Table 1: Training Data Set

Country's forum website	No. of pages	Pages selected	No. of selected texts
www.topix.com/forum/world/nigeria	35	1	30
www.topix.com/forum/world/ghana	10	1	45
www.topix.com/forum/world/liberia	4	1	10
www.topix.com/forum/world/cameroon	4	1	15
www.topix.com/forum/world/sierra-leone	4	1	15
Total no. of Texts			115

Table 2: Validation Data Set

first page of each country's website was carried out; all texts which met the conditions of at least twenty words with evidence of affiliation to the relevant country, but which had not been selected previously were selected. The one hundred and fifteen texts selected (as the validation set) are as shown in Table 2.

3.3.2 Sampling Technique for the Electronic Scam Data

A multistage sampling technique was also used for this population. The scam baiting sites selected for the study, the total number of electronic scam mails baited in each and the sample size are shown in Table 3.

Scam Baiter's website	No. of scam mails	No. selected
www.419eater.com	133	53
www.419hell.com	30	12
www.419baiter.com	19	8
www.scamorama.com	598	239
www.ebolamonkeyman.com	24	10
www.monkeyspit.net/inbox/	7	3
www.sweetchillisauce.com/nigeria.html	41	16
www.whatsthebloodypoint.com	11	4
http://419.bittenus.com	10	4
Total	873	349

Table 3: Scam Baiters' Websites and Sample Sizes (Date: 14/3/2011)

3.4 Text Pre-processing and Processing

The corpora were subjected to pre-processing in order to put them in the format expected by the relevant software for text processing. The pre-processing tasks included deletion of e-mail headers, removal of control codes, text aggregation, and removal of non-ASCII characters.

Text processing was achieved by extracting predetermined linguistic features from the sampled texts using computer codes written by the researchers in the Python2 programming language, based on the natural language toolkit (NLTK) version 2.0. Some of the specific issues handled in the course of text processing were tokenization, part of speech tagging and linguistic feature extraction.

Extracted features were syntactic features comprising the twenty (20) most frequent function words in the *topix.com* corpus, twenty (20) most frequent function words in the scam mail corpus (out of which seventeen most frequent function words that are common to *topix.com* corpus and scam mail corpus were used for analysis). Idiosyncratic features comprising adverb-verb part of speech (POS) bigram distribution and article omission/inclusion distribution. Structural features comprising lexical diversity; and content specific features twenty (20) most frequent noun, adjective, verb and adverb unigrams in the *topix.com* corpus, twenty (20) most frequent noun, adjective, verb and adverb unigrams in the scam mail corpus (out of which thirteen most frequent content words that are common to *topix.com* corpus and scam mail corpus were used for analysis).

The decision to extract twenty most frequent features (function word, noun, adjective, verb and adverb unigrams) was a result of a prior experiment by the researchers which showed that the summation of the frequencies of occurrence of the twenty most frequent features accounted for at least 60% of the cumulative frequency of all features extracted in each case.

3.5 Data Analysis

Although academic researchers have tended to favour the use of Support Vector Machines (SVMs), there is still a division on the choice of the best machine learning method, particularly in the anti-spam community (Sculley and Wachman, 2007). Also in the words of Witten and Frank (2005):

> *Experience shows that no single machine learning scheme is appropriate to all machine learning problems. The universal learner is an idealistic fantasy. Real datasets vary and to obtain accurate models, the bias of the learning algorithm must match the structure of the*

	TP Rate	FP Rate	Precision	Recall	F-score	ROC Area
NotNGGH	0.630	0.279	0.689	0.630	0.659	0.68
NGGH	0.721	0.370	0.665	0.721	0.692	0.68
Weighted Average	0.676	0.325	0.677	0.676	0.675	0.68

Table 4: Detailed Performance of the NigGh_Others Model

> *domain; machine learning like data mining is an experimental science.* (Witten and Frank: p. 365).

Thus, an experiment was carried out to determine the most precise machine learning algorithm for the study data set using the experimenter view of the open source Waikato Environment for Knowledge Analysis (WEKA) data mining software. Based on the result of the experiment, the Instance Based K-nearest (IBK) Neighbour algorithm was found to be the most precise in terms of accuracy and kappa statistic. Hence, the Instance Based K-nearest (IBK) Neighbour algorithm was used for the classification of the study's data sets based on a proposed hierarchical modelling approach.

3.5 Proposed Hierarchical Modelling Approach

This is a recursive approach proposed by the researchers based on the fact that human languages (and variants of a language) are not exclusively independent of one another (Bird, Klein and Loper, 2007), and on our understanding of the linguistic similarities among the English language variants of the five African countries in the study as revealed in Opesade, Adegbola and Tiamiyu (2013). Although each variant of the English language in the five countries differed from one another, there were still similarities among different pairs. For example, the Nigerian variant was found to be closest to the Ghanaian, followed be Sierra Leonean and Liberian and lastly Cameroonian.

In the proposed approach, binary classification tasks were performed such that the Nigerian and Ghanaian sub-corpora were first treated as one and separated from the other three, after which the two sub-corpora were classified into their respective country classes (Nigeria or Ghana). This was done because of the discovered similarity between the Nigerian and Ghanaian variants.

4. Presentation of Results

Research Question 1: What is the performance of the (NigGh_Others) model that classifies Nigerian and Ghanaian texts from those from the other three countries (Liberia, Cameroon and Sierra Leone)?

Based on 10-fold cross validation repeated ten times, the model's accuracy and kappa statistic were 67.619% and 0.3518 respectively. The detailed performance of the Instance based (IBK) model that classified training data set of Nigerian and Ghanaian linguistic origin from those of the three other countries is as shown in Table 4.

The precision values for the Nigeria/Ghana class (NGGH), others (NotNGGH) were 0.665 and 0.689 respectively, with a weighted average of 0.677. The model's recall values for the Nigeria/Ghana (NGGH), others (NotNGGH) were 0.721 and 0.63 respectively and with a weighted average of 0.676. The F-measures for the Nigeria/Ghana class (NGGH) and others (NotNGGH) were 0.692 and 0.659 respectively. The confusion matrix of the classification (Table 5) shows that four hundred and fifty-nine (459) out of seven hundred and twenty-eight (728) others (NotNGGH) texts were classified correctly, giving an accuracy value 63.0%.

NotNGGH	NGGH	Classified as	Accuracy (%)
459	269	NotNGGH	63.0
207	**535**	NGGH	72.1

Table 5: Confusion Matrix of the NigGh_Others Model using Topix_training Set

Five hundred and thirty-five (535) out of seven hundred and forty-two (742) Nigerian/Ghanaian (NGGH) texts were classified correctly, giving an accuracy value 72.1%. The NigGh_Others model was thereafter used to predict the sources of two hundred and fifty-three anonymous electronic scam mails as Nigerian/Ghanaian or other (Liberian, Cameroonian, Sierra-Leonean). The prediction is as shown in Table 6.

Predicted Class	Count	Percentage classified (%)
Nigerian/Ghana	119	47.04
Others (Liberia, Cameroon, Sierra Leone)	134	52.96
Total	**253**	**100**

Table 6: NigGh_Others model's Prediction of Electronic Scam Mails Classes

About forty-seven percent (47.04%), that is, one hundred and nineteen (119) of the scam mails were classified as being Nigeria/Ghana while the remaining 52.96% was classified as being Others (Liberia, Cameroon, Sierra Leone).

Research Question 2: What is the performance of the model (Nig_Gh model) that classifies Nigerian and Ghanaian texts only?

Texts from countries other than Nigeria and Ghana were removed from the training and Topix_test set, also scam mails classified as being from countries other than Nigeria and Ghana were excluded. The remaining texts were from two classes, that is, the Nigerian and Ghanaian class only. Accuracy and kappa statistic of the model were 67.66% and 0.3359 respectively. The detailed performance of the Nig_Gh model based on 10-fold cross validation is as shown in Table 7.

The precision values for the Ghanaian and Nigerian sub-corpora were 0.626 and 0.712 respectively, with a weighted average precision of 0.675. The model's values of recall for the Ghanaian and Nigerian sub-corpora were 0.603 and 0.732 respectively, with a weighted average recall of 0.677.

	TP Rate	FP Rate	Precision	Recall	F-score	ROC Area
NG	0.732	0.397	0.712	0.732	0.722	0.658
GH	0.603	0.268	0.626	0.603	0.614	0.658
Weighted Average	0.677	0.342	0.675	0.677	0.676	0.658

Table 7: Detailed Prediction Performance of the Nig_Gh Model

The F-measures for the Ghanaian (GH) and Nigerian (NG) classes were 0.614 and 0.722 respectively. Table 8 shows the confusion matrix of the classification (Nig_Gh) model.

Nigeria	Ghana	Classified as	Accuracy (%)
311	114	Nigeria	73.2
126	**191**	Ghana	60.3

Table 8: Confusion Matrix of the Nig_Gh Model on Training Set

The confusion matrix shows that three hundred and eleven (311) out of four hundred and twenty-five (425) Nigerian texts were classified correctly, giving an accuracy value of 73.2% for the Nigerian sub-corpora; while one hundred and ninety-one (191) out of three hundred and seventeen (317) Ghanaian texts were classified correctly, giving 60.3% accuracy value. The Nig_Gh model was thereafter used to predict the sources of the one hundred and nineteen (119) anonymous electronic scam mails classified as Nigerian/Ghanaian in Table 5. The prediction is as shown in Table 9.

Predicted Class	Count	Percentage (%)
Nigerian	73	61.3
Ghanaian	46	38.7
Total	**119**	**100**

Table 9: Nig_Gh model's Prediction of Electronic Scam Mails Classes

Out of the one hundred and nine (119) scam mails classified to be from a combination of Nigerian and Ghanaian texts in Table 5, the Nig_Gh model predicted seventy-three (73) to be from Nigeria and forty-six (46) to be from Ghana.

Research Question 3: What percentage of electronic scam mail are classified to be Nigerian relative to the other four countries, based on linguistic cues?

Table 10 shows the percentage of scam mail classified to be from Nigeria relative to the other four countries. Seventy-three (73) out of the 253 scam mail in the study were classified to be from Nigeria. This shows that about 28.85% of the scam mails are classified to be from Nigeria based on linguistic cues.

4.1 Model Validation

The two-level model approach was validated on the one hundred and fifteen (115) *topix* validation data set. The accuracy and kappa statistic were 60.87% and 0.2153 respectively for the NigGh_Others model and 70.67 % and 0.3774 respectively for the Nig_Gh model**.** The weighted F-Measures were 0.618 and 0.716 for the NigGh_Others and Nig_Gh models respectively.

Predicted Class	Count	Percentage (%)
Nigerian	73	28.85
Ghanaian	46	18.18
Others (Cameroon, Liberia, Sierra Leone)	134	52.96
Total	**253**	**100**

Table 10: Nig_Gh Model's Prediction of Electronic Scam Mails' Classes

5. Discussion of Findings on the Research Questions

Using the two-level modelling approach, 28.85% of anonymous electronic scam mails were classified as being from Nigeria. This showed that among five countries, about one-third of the two hundred and fifty-three electronic scam mail in the study were predicted to be from Nigeria. This result is mid-way between the findings of Longe and Osofisan (2011) whose analysis of electronic mails provided results that deviated from the generally held beliefs and cast some shadows on widely held opinions on the origins of 419 mails and that of Edelson (2003) who reported that 67% of the scam e-mails were from Nigeria. The finding of this study therefore, supports the third view of the controversial views on the acclaimed source of electronic scam mail. This third view as submitted by Smith *et al.* (2001) is that the term "Nigerian Advance Fee Fraud" is only partially accurate and the problem is truly one of international dimension with victims and offenders being located across the globe. However, Nigeria may be one of the countries where scam mail authoring is prominent, as informed by a relatively high percentage accrued to the country among four other countries.

The percentage of scam mail classified to be Nigerian was higher than average, although not as high as one would expect as it has been believed generally that Nigeria was responsible for the crime. The implication of this is that electronic scam mail problem cannot be pinned down to Nigeria. However, it is possible that Nigeria is one of the countries that are prominent in authoring such mails. The performance of our model despite the few words contained in the comments of the study corpora shows that the use of linguistic cues in authorship profiling of anonymous texts is promising. Finally, the improvement in model performance when the similarity between Nigerian and Ghanaian texts was put into consideration implies that linguistic similarities between variant of language in consideration should be resolved in order to improve classification model performance.

5.1 Limitations of the Study

The availability of a corpus of electronic scam mail with authenticated country sources could have been more appropriate for the study, in the absence of this, the available corpus of *www.topix.com* was used as a surrogate. Other limitations include the relatively low number of words in the comments of the training and validation dataset, natural language processing applications, especially for the purpose of authorship attribution perform better with large text sizes. Also, the study modelled the specific linguistic features of the Nigerian variant because of the motivation for the study; however, modelling the linguistic features of each of the dialects under consideration could have produced a better performance of the model.

5.2 Recommendations

Based on the finding of this study, it is recommended that researchers in technology should exploit the results of studies in the humanities in general and languages of communication specifically to enable the optimization of technology to solve human problems such as the forensic determination of the origin of scam mail through linguistic analyses. There should be more research in the creation of text corpora in Nigeria. This is necessary because the availability of corpora will go a long way in helping researchers to carry out statistical language modelling of Nigerian text more readily. Also law enforcement agencies of each country, in particular Nigeria, should create a database of investigated fraudulent mails, with contents stored and tagged appropriately with the criminals' profiles. Such data can be useful in investigating transnational digital criminal offences by applying the proposed statistical language modelling technique.

Finally, it is recommended that governmental, non-governmental and education sectors in Nigeria should provide seminars and training on more profitable use of the computer technologies including the Internet and also on the evil of electronic scamming to the Nigerian youth. The information age and its numerous enabling technologies have come to stay, deliberate introduction of positive use of these devices will reduce their possible negative uses.

6. Conclusions

Based on the findings of the study, it could be concluded that linguistic cues have potentials of being used for investigating transnational digital breaches if properly exploited. More research effort in this area, particularly in the deliberate storage of annotated electronic texts from different countries of the world could serve as a useful resource if a need arises to profile the country source of a controversial anonymous text. It could also be concluded that electronic scam mail problem cannot be pinned down to Nigeria as believed generally, though Nigeria could be one of the countries that are prominent in authoring such mails as revealed by the study.

7. Bibliographical References

Adetugbo, A. (1979). Appropriateness of Nigeria English. In E. Ubahakwe (Ed.), Varieties and Functions of English in Nigeria: Selections from the Proceedings of the Ninth Annual Conference of the Nigerian English Studies Association. Ibadan, Nigeria: African Universities Press, pp. 167-183.

Alshalan, A. (2006). Cyber-crime fear and victimization: an analysis of a national survey. Doctoral Thesis. Mississippi State University, Department of Sociology, Anthropology, and Social Work.

Banjo, A. (1979). Beyond intelligibility: A presidential address. In E. Ubahakwe (Ed.), Varieties and Functions of English in Nigeria: Selections from the Proceedings of the Ninth Annual Conference of the Nigerian English Studies Association. Ibadan, Nigeria: African Universities Press, pp.7-13.

Bayart, J., Ellis, S. & Hibou, B. (1999). The Criminalization of the State in Africa. Bloomington: Indiana University Press.

Bird, S., Klein, E & Loper, E. (2007). Natural Language Processing in Python. USA: O'Reilly Media.

Choo, K.R. (2011). The cyber threat landscape: challenges and future research directions. Computers and Security, 30, pp. 719 – 731.

Cruz-Ferreira, M. & Abraham, S.A. (2006). The Language of Language-core Concepts in Linguistic Analysis. 2nd ed. Singapore: Prentice Hall Pearson.

De Vel, O. , Anderson, A., Corney, M. & Mohay, G. (2001). Mining e-mail content for author identification forensics. Special Interest Group on Management of Data (ACM SIGMOD) Record, 30(4), pp. 55-64.

Dyrud, M.A. (2005). I Brought You a Good News: An Analysis of Nigerian 419 letters. In Proceedings of the 2005 Association for Business Communication Annual Convention. Retrieved Feb. 12, 2010, from http://www.businesscommunication.org/conventionsNew/proceedingsNew/2005New/PDFs/07ABC05.pdf.

Edelson, E. (2003). The 419 scam: information warfare on the spam front and a proposal for local filtering. Computers and Security, 22 (5), pp. 392-401.

Egbe D.I. (1979). Spoken and written English in Nigeria. In E. Ubahakwe (Ed.), Varieties and Functions of English in Nigeria: Selections from the Proceedings of the Ninth Annual Conference of the Nigerian English Studies Association. Ibadan, Nigeria: African Universities Press, pp. 86-106.

Iqbal, F., Hadjidj, R., Fung, B.C.M. & Debbabi, M. (2008). A Novel Approach of Mining Write-Prints for Authorship Attribution in E-mail Forensics. In 2008 Digital Forensic Research Workshop. Elsevier Ltd. Retrieved Nov. 16, 2009, from www.elsevier.com/locate/diin. 2008.05.001.

Jowitt, D. (1991). Nigerian English Usage: An Introduction. Nigeria: Longman.

Juola, P. (2007). Future trends in authorship attribution. International Federation for Information Processing, 24(2). pp. 119-132.

Koppel, M., Schler, J., Argamon, S. & Messeri, E. (2006). Authorship Attribution with Thousands of Candidate Authors. In: Proceedings of the 29th Annual International ACM SIGIR (Special Interest Group on Information Retrieval) Conference on Research and Development in Information Retrieval. Seattle, USA: ACM, pp.659-660.

Koppel, M., Schler, J., & Argamon, S. (2009). Computational methods in authorship attribution. Journal of the American Society for Information Science and Technology, 60(1), pp. 9-26.

Kujore, O. (1985). English Usage: Some Notable Nigerian Variations. Nigeria: Evans Brothers Nigeria Publishers Limited.

Longe, O. & Osofisan, A. (2011). On the Origins of Advance Fee Fraud Electronic Mails: A Technical Investigation Using Internet Protocol Address Tracers. The African Journal of Information Systems, 3(1), 17-26.

Opesade, A., Adegbola, T. & Tiamiyu, M. (2013). Comparative analysis of idiosyncrasy, content and function word distributions in the English language variants of selected African countries. International Journal of Computational Linguistics Research, 4(3), pp. 130-143.

Poplack, S. (1993). Variation theory and language contact. In D.R. Preston (Ed.), American Dialect Research. USA: John Benjamins Publishing Co, pp. 251-286.

Reith, M., Carr, C. & Gunsch, G. (2002). An examination of digital forensic models. International Journal of Digital Evidence, 1(3), pp. 1-12.

Rosenfeld, R. (2000). Two Decades of Statistical Language Modelling: Where Do We Go From Here? In: Proceedings of the European Conference on Speech Communication and Technology. Retrieved Nov. 20, 2010, from citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.140.5518.

Sculley, D. & Wachman, G.M. (2007). Relaxed Online SVMs for Spam Filtering. In: SIGIR 2007 Proceedings. USA: ACM, pp. 415-422.

Smith, R.G., Holmes, M. N. & Kaufmann, P. (2001). Nigerian advance fee fraud. International Society for the Reform of Criminal Law. Retrieved Feb. 14, 2010, from http://www.isrcl.org/Papers/Nigeria.pdf.

State Department Publication. (1997). Nigerian advance fee fraud. Retrieved 16 June, 2009, from http://www.state.gov/documents/organization/2189.pdf.

Torosyan, A. (2003). Cyber crime programs by state and local law enforcement: a preliminary analysis of a national survey. Doctoral Thesis. California State University, Department of Criminal Justice.

Wardhaugh, R. (1994). An Introduction to Sociolinguistics. 2nd. ed. U.K: Blackwell Oxford.

Witten, I.H. & Frank, E. (2005). Data Mining: Practical Machine Learning Tools and Techniques. 2nd ed. USA: Morgan Kaufmann publishers.

Android Malware Classification through Analysis of String Literals

Richard Killam, Paul Cook, Natalia Stakhanova
Faculty of Computer Science
University of New Brunswick
richard.killam@unb.ca, paul.cook@unb.ca, natalia.stakhanova@unb.ca

Abstract

As the popularity of the Android platform grows, the number of malicious apps targeting this platform grows along with it. Accordingly, as the number of malicious apps increases, so too does the need for an automated system which can effectively detect and classify these apps and their families. This paper presents a new system for classifying malware by leveraging the text strings present in an app's binary files. This approach was tested using over 5,000 apps from 14 different malware families and was able to classify samples with over 99% accuracy while maintaining a false positive rate of 2.0%.

Keywords: Mobile Malware, Android, String analysis

1. Introduction

The rising popularity of the Android platform has led to an increase in observed malware, as much as 391% from 2013 to 2014 (Pulse Secure Mobile Threat Center, 2015). This unprecedented increase is often associated with the inherit openness of the Android platform, and the corresponding lack of security measures to prevent potential abuse.

Consequently, a significant amount of research attention has focused on exploring the nature of malware applications and developing techniques for their detection. A wide range of malware detection approaches have been introduced, from generic methods such as RiskRanker (Grace et al., 2012a), DroidRanger (Zhou et al., 2012b), and Drebin (Arp et al., 2014), to more specialized methods offering detection of repackaging (Crussell et al., 2013; Gonzalez et al., 2015; Zhou et al., 2013) and privacy violations (Enck et al., 2010; Zhang et al., 2013). Keeping up with the growing prevalence of Android malware, the existing studies focus on either a complex multi-feature analysis capable of providing insight into a malware sample (e.g., DroidSafe (Gordon et al., 2015)), or a large-scale binary classification identifying malicious and benign samples during the triage stage.

With the increasing amount and complexity of malware, in triaging it is beneficial to employ methods that do not rely on sophisticated analyses, and are capable of providing insight into the malicious functionality represented by a malware app.

Since core functionality is shared by samples of the same malware family, identification of the malware family that a sample belongs to is the first step in understanding and assessing malware impact and potential damage.

In this work, we propose a machine-learning based approach for the detection of Android malware through analysis of unreferenced, hidden strings present in Android executables. In general, strings have multiple uses in Android binaries, including referencing code components based on class or methods' names. Prior research has focused mostly on analysis of the code portions of Android executables (i.e., .dex files) referenced through corresponding strings. Since this is typical and therefore expected code invocation method, it is commonly employed in reverse engineering of Android applications (commonly referred to as apps). Leveraging this, malware authors often avoid detection by doing indirect code invocation by placing code in unreferenced strings. In this work, we propose to explore these components. Specifically, we investigate the role of strings found within an Android executable, but not referenced by the code.

We evaluate the proposed method with a dataset made available through The Android Malware Genome project (Zhou and Jiang, 2012), and a collection of over 4,000 benign apps retrieved from Google Play market. Our findings using this novel approach based on unreferenced strings show that this is an efficient alternative to prior approaches to malware classification. Our best approach achieves 99.3% accuracy, and a false positive rate of just 0.5%, for malware detection, and 99.2% accuracy with a false positive rate of 2.0% for malware family classification.

We further contrast our approach against the one using all strings. Our experiments show that using only a small set of unreferenced string features, we are able to effectively identify a suspicious app maintaining the same high accuracy as with all strings.

The rest of this paper is organized as follows: we present related work in Section 2. Our system design is described in Section 3. A detailed description of the features used for classification are outlined in Section 4. We describe our experimental setup in Section 5. We present results in Section 6. Finally, we conclude our work in Section 7.

2. Related Work

In recent years the area of mobile security has seen extensive growth and improvement. A broad overview of characteristics of mobile malware, and approaches to its detection, has been given by Zhou et al. (2012), Polla et al. (2013), and Alzahrani et al. (2014).

The resource-constrained environment of mobile platforms presents significant challenges to the detection of malware. As a result, features that detection techniques rely on play a critical role for the accuracy of malware detection. The features commonly used in existing classification studies can be grouped into two categories: dynamic features derived from an app's behaviour during runtime, and static features, extracted from an app itself.

Systems that use dynamic analysis, RiskRanker (Grace et al., 2012b), focus on behaviour while running, typically monitoring s made by the app. CopperDroid (Tam et al., DroidScope (Yan and Yin, 2012) are example niques that utilize dynamic features at a host system calls). Arora et al. (2014) focused on net features extracted from app behaviour.

Techniques that employ static analysis target th are packaged with an application. This eliminat to execute the app in order to detect any ma tent. The majority of systems that utilize stat have focused on two types of packaged files: An ifest.xml — which holds the app's metadata – cutable files. App permissions contained in An ifest.xml have been explored by several studi ing DroidRanger (Zhou et al., 2012b), Drebin (2014), and Auditor (Talha et al., 2015).

Static features extracted from executables often ditional preprocessing, and are commonly used in the form of opcode or bytecode n-grams. Fc Wolfe et al. (2014) analyze the bytecode n-gram from Android binaries. Juxtapp (Hanna et al., 2013) evaluates code similarity based on opcode n-grams extracted from selected packages of the disassembled .dex file. Similarly, DroidMOSS (Zhou et al., 2012a) evaluates app similarity using fuzzy hashes constructed based on a series of opcodes. DroidKin (Gonzalez et al., 2014) detects unique apps by analyzing the similarity of opcode n-grams and metadata of apps.

In contrast to these previous studies, we propose a novel form of static analysis based specifically on features derived from the unreferenced strings contained within an executable.

3. Android String Analysis

An Android application package (APK) file, is a compressed folder that contains a variety of files including an executable .dex file; a manifest file (AndroidManifest.xml) that describes the content of the package; and resources files (e.g., image and sound files). The Dalvik executable file, or simply .dex file, is a binary that results from compiling the app's Java source code. As illustrated by Figure 1, this file is partitioned into a number of sections that describe aspects of the structure of the file.

Among these sections are several identifier lists that contain offsets pointing to the corresponding entries in the data section. As such, the string identifiers list (i.e., the string_ids section) provides offsets to all strings used by the .dex file. These strings are used for internal naming — e.g., class, field, or local variable names — and to reference constant objects specified in the source code (e.g., string literals). The other identifier sections — type_ids, proto_ids, field_ids, method_ids, and class_defs — may also contain references to the string identifiers list. For example, a class named `myclass` will have a corresponding entry in the string identifier section pointing to the actual string *myclass*.

This structure is defined in a formal specification of the layout of .dex executable files guiding the development of

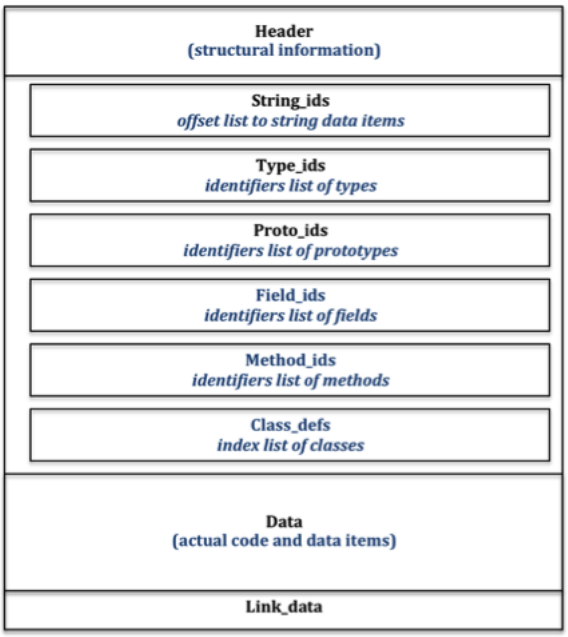

Figure 1: The structure of a DEX file.

Android apps.[1] Although app development must follow the suggested guidelines, there are a number of techniques that enable parts of code to be hidden in an Android executable (Apvrille, 2012; Apvrille, 2013). The main premise of these techniques is to place the code inside the data section while avoiding it being referenced by class and method index lists. Since these lists are typically used to invoke the methods and classes, this prevents reverse engineering of an app and allows malware apps to bypass anti-malware scans.

The majority of prior research has focused on the code section of the officially documented .dex structure. In contrast, we explore the text components. As such we differentiate between strings that are pointed to by any of the other identifier sections, denoted as "referenced strings", and all other strings referred to as "unreferenced strings". Referenced strings can be viewed as a part of functional app code. They are linked to other identifier lists and thus, for example, can represent classes, methods, or data structures. On the other hand, the data section also contains "unreferenced" strings that are only referenced by the string offset list, and thus carry textual information, e.g., string literals. In prior work on malware analysis this part of the code has not carried the same weight as the functional portion. However, these unreferenced strings often carry hidden or interesting information. For example, hardcoded URLs and email addresses (represented as strings) are common in malware apps, and would occur amongst the unreferenced strings. As such, analysis of unreferenced strings can potentially indicate malicious activity embedded in Android apps.

In this paper we investigate the impact of unreferenced strings found within the Android executable on the tasks

[1] `https://source.android.com/devices/tech/dalvik/dex-format.html`. Accessed: 2016-02-11

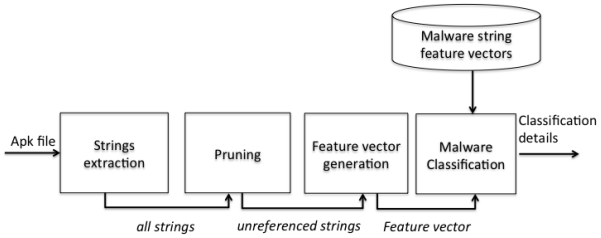

Figure 2: The sequence of analysis in the proposed approach.

of identifying malicious Android apps, and classifying Android apps with respect to malware family.

3.1. String analysis

To investigate the role of unreferenced strings in malware analysis, the proposed system encompasses several steps: string extraction, pruning, feature vector generation, and finally assessment, as shown in Figure 2. The initial steps of string extraction and pruning primarily focus on preprocessing activities that are necessary to allow our system to carry out feature vector generation. These vectors then serve as the basis for the following classification and analysis.

In the *strings extraction* step, for each app under analysis, our system recursively unpacks and extracts all .dex files found in both the root APK, as well as within all sub-APKs. All of these .dex files contribute to the final string data associated with the root app. The strings of each .dex file are extracted by traversing the string_ids list and collecting all strings present in data section.

The *pruning* step then prunes these strings to only include the unreferenced strings. This is accomplished by traversing the 5 other identifier lists — type, proto, field, method, and class — and then removing strings that are referenced by these lists. The resulting unreferenced strings therefore consist solely of non-executable code, such as string literals and local variable names, as well as strings from hidden executable parts of code not referenced through identifier lists. These remaining strings then form the basis for feature vectors.

To generate feature vectors in the *feature vector generation* step, we consider word-level string n-gram features (described in the following section) derived from the unreferenced strings. As a point of comparison, we also consider feature vectors generated from all strings (i.e., referenced and unreferenced strings). To form these feature vectors, we simply omit the pruning step.

In contrast to our proposed approach, malware analysis is typically conducted at the bytecode or opcode level. Opcodes are generally beneficial in representing the low-level semantics of the code, while bytecode is seen as the complete representation of the code at a low-level. As a further important point of comparison, we therefore also experiment with opcode and bytecode n-grams (also described in the Section 4.). For these opcode and bytecode features, the strings extraction and pruning phases do not apply.

Feature vectors are generated for each app in a large collection of Android apps, each of which is known to be either a malicious app, or a benign (non-malware) app. In the case of malware, the malware family is also known. These feature vectors are used to train supervised classifiers to identify malicious Android apps, and classify Android apps according to their malware family. After training, in the *malware classification* step the malware status and family of a new app is assessed based on its classification under the previously-trained classifier model.

4. String features

To preserve the semantics of the original strings, the string features used in the proposed approach were extracted at the word level. They were then processed into n-grams of varying lengths. An n-gram is a contiguous sequence of n items. In the case of word level n-grams these items are words in a sentence. For example, given the sentence *Good morning John Doe.*, three 2-gram tokens can be made:

1. *Good morning*
2. *morning John*
3. *John Doe.*

Note that the tokenization used here was to split a string based on whitespace; therefore in this case the period at the end of the sentence is a part of the word *Doe.*

In the feature vector generation process, each extracted string was written onto a separate line, so that these word level n-grams could be line bounded. This was done to ensure that n-grams did not contain words from different lines. Since the strings are held in lexicographical order, adjacent strings are not necessarily related. As such, n-grams spanning multiple lines would contain word sequences that are not associated, that is, words that are not a part of the same original string. These n-grams would contain meaningless relationships, and would thus add noise to the feature set, which could hinder classification.

4.1. String Feature Extraction

Each app was represented using both token (n-gram) frequencies and token frequency-inverse document frequency (tf-idf) weights. Frequency vectors contain the number of times a term (i.e., an n-gram) occurs in a given document (where in our case a "document" is the collection of strings extracted from a given app); tf-idf assigns lower weights to terms that occur in many documents, and is calculated as follows:

$$\text{tf-idf}_{i,d} = f_{i,d} * \log\left(\frac{N}{n_i}\right) \quad (1)$$

where $f_{i,d}$ is the frequency of term i in document d, N is the total number of documents, and n_i is the number of documents that term i occurs in at least once.

These tokens were extracted by grouping the strings on each line into word level n-grams, with n ranging from 1 to 4. Any lines that had fewer than n words were ignored for that feature set. The analysis of string sizes across our dataset showed that only a few strings had more than 4 words (Figure 3). As such in this work we capped n-gram length at 4.

Table 1: Examples of n-gram features. <LB> is the line boundary token.

Line Text	1-grams	2-grams	3-grams	4-grams
mThread=	mThread=	<LB> mThread= mThread= <LB>	<LB> mThread= <LB>	
wrote final 256;	wrote final 256;	<LB> wrote wrote final final 256; 256; <LB>	<LB> wrote final wrote final 256; final 256; <LB>	<LB> wrote final 256; wrote final 256; <LB>

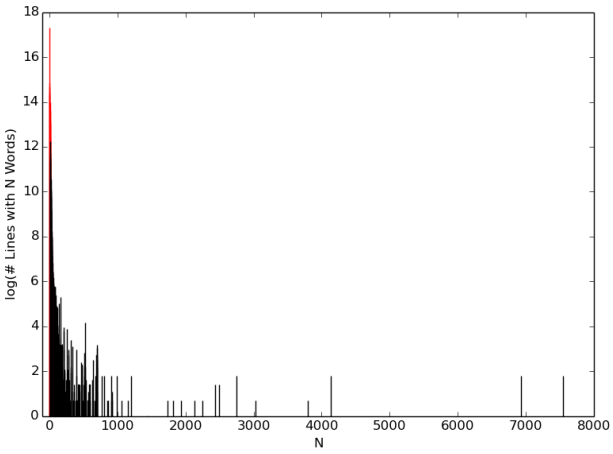

Figure 3: The distribution of string lengths in the dataset.

Table 2: Distribution of Android APK files across malware families

	Malware Family	File Count
1	ADRD	22
2	AnserverBot	187
3	BaseBridge	122
4	DroidDreamLight	46
5	DroidKungFu1	34
6	DroidKungFu2	30
7	DroidKungFu3	309
8	DroidKungFu4	96
9	Geinimi	69
10	GoldDream	47
11	KMin	52
12	Pjapps	58
13	YZHC	22
14	Other	166
15	Benign GooglePlay apps	4,574
	Total	5,834

If $n > 1$ then a unique line boundary token was added to the start and end of each line. This ensured that all lines contained at least 3 words (including the line boundary tokens), thereby reducing the number of lines that were ignored for having fewer than n words, and therefore making more information available in the feature vectors. Adding line boundary tokens also serves to identify the words and n-grams that occur at the beginning and end of strings. This could be potentially important information in that a string that begins with, for example, *warning* or *error* could mean something very different than a string that simply contains these words.

Line boundary tokens were not added to the 1-gram features. The frequency of these special line boundary tokens in this case would be equivalent to counting the number of strings in the app. As this information is not directly encoded by the other n-gram features, it was also avoided here.

An example of the proposed feature extraction method is shown in Table 1. Note the following important properties:

- There are no <LB> tokens for the 1-gram feature sets.
- There were no 4-gram tokens for the first string, "mThread=". This is because, even with the line boundary tokens, the string only contained 3 words. As such, it was ignored for not having at least $n = 4$ words in it.
- The words in the 2 and 3-gram tokens were in the same order in which they appeared in the original string, and the words were always adjacent. That is, the 2-gram feature set did not contain tokens such as "final wrote" or "wrote 256;".

5. Experimental setup

5.1. Data

To evaluate the performance of the proposed malware analysis approach we employed a standard benchmark dataset produced by The Android Malware Genome project (Zhou and Jiang, 2012). This dataset consists of 1,260 malware apps. We also required a collection of benign (i.e., non-malware) apps for evaluation purposes. We performed a large-scale study of the top Android applications retrieved from Google Play market between September 2014 and January 2015. These apps were inspected by ESET antivirus scanner to detect the presence of malware. All malicious apps were removed from the dataset. This resulted in 4,574 benign (i.e., non-malware) GooglePlay apps which were used in this study.

The apps retrieved from Google play market and The Android Malware Genome project were used for classification experiments. This dataset of 5,834 apps is detailed in Table 2.

Some families in this dataset ended up with only a few samples, making classification and validation of results difficult. As such we grouped low frequency families into an "Other" class. The app distribution among the families had a convenient gap between families with 16 and 22 samples. As such, all families that had fewer than 17 files were considered to be low frequency, and placed in the 'Other' class.

Thereby resulting in 15 distinct malware families, including the benign GooglePlay apps.

5.2. Feature extraction

Opcodes for each DEX file were extracted using dexdump, a disassembler tool that is a part of the Android SDK. Opcode operands were discarded as their variability typically results in noise and excessively sparse feature vectors which makes classification more difficult.

Each app was represented using opcode n-grams, where n ranged from 1 to 4, the same range of n considered for string n-grams. Opcode n-grams were DEX file bounded (as opposed to line bounded in the case of string n-grams). A special DEX file boundary token was included for the opcode n-grams for $n > 1$, for similar reasons as the inclusion of the line boundary token in the case of string n-grams.

The bytecode features were obtained by reading each byte of each DEX file. Since DEX files start with the magic number "dex",[2] thus acting as built-in boundary tokens, additional boundary tokens were not used. Each app was represented using bytecode n-grams, where n ranged from 1 to 2.

5.3. Classification and Baseline

We implemented the proposed approach in Python, using the scikit-learn library (Pedregosa et al., 2011). In preliminary experiments we considered a variety of classifiers including k-nearest neighbors, multinomial naive Bayes, logistic regression, and a linear support vector machine. The linear SVM (Fan et al., 2008) had the best performance, or close to it, in terms of both accuracy and false positive rate on malware detection and family classification. We therefore focus on, and report results for, the linear SVM classifier for the remainder of the paper.

In the case of the datasets used in this study, and in most application marketplaces, benign applications outnumber malicious applications. In a case such as this where the classes (benign and malicious) are imbalanced, a very naive strategy of simply selecting the most frequent class can perform very well, particularly in terms of accuracy. We therefore also report a majority class baseline — referred to as "Baseline" — that classifies all samples as the most common class in the training data. In our experiments this strategy will classify all samples as benign. This baseline provides a reference against which we can interpret the results of other methods. Crucially, it is not the only approach against which we compare our string-based classification approaches; we also consider approaches based on opcodes.

6. Experimental Results

In our experimental study we focused on analysis of:

1. Binary classification: classification of a sample as either a malicious or benign app. This type of classification is also referred to as malware detection.
2. Malware family classification: where the samples were classified as one of 15 different classes, 14 of which are malware families and 1 class representing benign apps. These families, and the number of files in them, are listed in Table 2.
3. Unreferenced strings vs all strings: The classification performance using unreferenced strings was contrasted against that using all strings (i.e., referenced and unreferenced).
4. Classification with selected string features: classification of apps based on selected, highly-informative features.

All classification experiments were repeated 5 times, each using stratified 5-fold cross validation.

6.1. Unreferenced strings vs all strings

We compared the performance of the linear SVM when using all of the strings versus using only the unreferenced strings. These results are shown in Table 3. The results are, overall, quite similar. For example, the best accuracy for malware detection ("Binary") is 99.5% for all strings, and 99.3% for unreferenced strings. For malware family classification there is a similarly small difference in terms of the best accuracy for the all strings and unreferenced strings approaches. That the all strings approach does slightly better does not come as a surprise; retaining all strings provides more information to the classifier. However, with unreferenced strings, we are able to drastically cut the total number of strings that are examined (from roughly 127 million to 46 million) while maintaining a similar accuracy.

The performance results of the linear SVM classifier using only the unreferenced strings for malware detection and malware family classification are shown in Table 4. Unsurprisingly, all n-gram classification strategies performed significantly better than the baseline in all cases, with the exception of the false positive rate for malware detection (Binary).

For both malware detection and family classification, we see an increase in performance for all metrics when increasing the size of the word grams from 1 to 2, and again when increasing from 2 to 3. This is likely caused by the increased contextual information that is carried by higher order n-grams. However, there is a clear drop in performance for 4-grams. Two factors could contribute to this. Data sparsity could be an important factor, because many 4-grams will be unique. Furthermore, the system ignores strings that contain fewer than n words after padding with line boundary tokens. The 4-gram strategy ignored over 30M lines throughout the dataset. One possibility to overcome this would be to pad lines with additional word boundary tokens; however, this would substantially increase the number of features in this model, making classification much more computationally intensive.

The best strategy for both malware detection and family classification was to use 3-gram word counts. This strategy was able to achieve near perfect ($> 99\%$) accuracy for both malware detection and malware family classification while maintaining a low false positive rate. Moreover, the

[2] `https://source.android.com/devices/tech/dalvik/dex-format.html` Accessed: 2016-02-11

Table 3: Accuracy (Acc), macro-averaged precision (Prec) and recall (Rec), and false positive rate (FPR) for malware detection (Binary) and malware family classification (Family) using a linear SVM with features based on all strings (All) and just the unreferenced strings (Unref).

	Strategy	Acc		Prec		Rec		FPR	
		All	Unref	All	Unref	All	Unref	All	Unref
Binary	1-gram word freq.	99.3%	98.8%	97.8%	96.5%	99.0%	98.1%	0.63%	0.99%
	2-gram word freq.	99.5%	99.2%	98.6%	98.0%	99.1%	98.5%	0.40%	0.56%
	3-gram word freq.	98.6%	99.3%	96.8%	98.1%	96.6%	98.6%	0.87%	0.50%
	4-gram word freq.	98.5%	98.6%	96.7%	96.9%	96.5%	96.7%	0.89%	0.85%
	1-gram word tf-idf	99.4%	99.3%	99.3%	99.4%	98.0%	97.2%	0.18%	2.8%
	2-gram word tf-idf	99.3%	99.2%	99.3%	99.3%	97.3%	97.2%	0.19%	2.8%
	3-gram word tf-idf	98.8%	99.2%	99.2%	99.3%	95.0%	96.9%	0.21%	3.1%
	4-gram word tf-idf	98.6%	98.6%	99.1%	99.1%	94.5%	94.5%	0.25%	0.24%
Family	1-gram word freq.	99.0%	98.5%	97.4%	96.7%	97.2%	96.5%	88.6%	2.7%
	2-gram word freq.	99.5%	99.0%	98.4%	98.3%	98.4%	97.5%	94.9%	1.6%
	3-gram word freq.	98.2%	99.2%	95.9%	98.6%	95.2%	98.0%	4.8%	2.0%
	4-gram word freq.	98.2%	98.4%	94.9%	96.2%	94.9%	95.5%	5.1%	4.5%
	1-gram word tf-idf	98.5%	96.6%	98.6%	98.7%	93.3%	94.4%	6.7%	5.6%
	2-gram word tf-idf	98.3%	98.7%	98.0%	98.6%	92.4%	94.9%	7.6%	5.1%
	3-gram word tf-idf	97.5%	98.4%	96.0%	97.9%	88.6%	93.2%	11.4%	6.8%
	4-gram word tf-idf	97.1%	97.1%	93.5%	93.5%	85.0%	85.0%	15.0%	15.0%

Table 4: Accuracy (Acc), macro-averaged precision (Prec) and recall (Rec), and false positive rate (FPR) for malware detection (Binary) and malware family classification (Family) using a linear SVM with features based unreferenced strings, bytecodes, and opcodes.

Strategy	Binary				Family			
	Acc	Prec	Rec	FPR	Acc	Prec	Rec	FPR
Baseline	78.4%			0.00%	78.4%			93.3%
1-gram word freq.	98.8%	96.5%	98.1%	0.99%	98.5%	96.7%	96.5%	3.5%
2-gram word freq.	99.2%	98.0%	98.5%	0.56%	99.0%	98.3%	97.5%	2.5%
3-gram word freq.	99.3%	98.2%	98.6%	0.50%	99.2%	96.2%	95.5%	2.0%
4-gram word freq.	98.6%	96.9%	96.7%	0.85%	98.4%	98.6%	98.0%	4.5%
1-gram word tf-idf	99.3%	99.4%	97.2%	2.8%	96.6%	98.7%	94.4%	5.6%
2-gram word tf-idf	99.2%	99.3%	97.2%	2.8%	98.7%	98.6%	94.9%	5.1%
3-gram word tf-idf	99.2%	99.3%	96.9%	3.1%	98.4%	97.9%	93.2%	6.8%
4-gram word tf-idf	98.6%	99.1%	94.5%	0.24%	97.1%	93.5%	85.0%	14.99%
1-gram bytecode freq.	85.6%	70.1%	70.7%	10.29%	74.8%	24.7%	21.4%	78.58%
2-gram bytecode freq.	92.3%	83.5%	83.0%	5.09%	83.0%	39.7%	29.7%	70.26%
1-gram bytecode tf-idf	79.0%	93.0%	3.14%	0.06%	78.4%	7.0%	6.8%	93.23%
2-gram bytecode tf-idf	94.1%	86.8%	86.0%	3.62%	82.2%	19.2%	14.5%	85.53%
1-gram opcode freq.	97.9%	95.0%	95.3%	1.38%	95.9%	84.7%	85.3%	14.71%
2-gram opcode freq.	98.1%	95.0%	96.4%	1.42%	96.7%	90.1%	87.9%	12.07%
3-gram opcode freq.	98.2%	95.0%	96.8%	1.39%	97.4%	94.0%	91.5%	8.52%
1-gram opcode tf-idf	98.5%	96.0%	97.3%	1.12%	95.7%	92.4%	77.0%	22.99%
2-gram opcode tf-idf	99.3%	98.1%	98.7%	0.52%	97.9%	95.2%	89.7%	10.32%
3-gram opcode tf-idf	99.2%	98.0%	98.4%	0.56%	97.9%	95.2%	90.7%	9.28%

accuracy, precision, and recall of this method is better than that for any of the classifiers using the more-conventional bytecode or opcode n-gram features.

6.2. Feature Selection

Since anti-malware vendors are forced to maintain large and ever-growing numbers of signatures for malware detection, it is important that any malware detection and classification methods be able to use a minimal number of features. As such, we tested the proposed method using only the 50 highest ranked 3-gram tokens, extracted from the unreferenced strings, for each family.

These 3-grams were ranked by training a linear SVM using the one-vs-rest classification strategy. Features that have high coefficients have more of an impact on how the linear SVM classifies a given sample. Therefore, the 50 tokens that had the highest coefficient for each family, are equivalent to the 50 highest ranked tokens for each of those families. We present examples of the top-5 ranked 3-grams for selected families in Table 5. By using only these 50 strings

Table 5: The 5 most important 3-grams when used to classify selected malware families. Note that <LB> is the line boundary token and that the whitespace between the words was removed to reduce memory consumption when training.

Class	Rank	Token
Benign apps	1	<LB> this <LB>
	2	<LB> accessFlags <LB>
	3	<LB> android.intent.action.VIEW <LB>
	4	<LB> string <LB>
	5	<LB> com.android.vending <LB>
DroidKungFu2	1	<LB> /system/etc/.rild_cfg <LB>
	2	<LB> /system/etc/.dhcpcd <LB>
	3	<LB> /WebView.db <LB>
	4	<LB> WebView.db.init <LB>
	5	<LB> /secbino <LB>
DroidKungFu3	1	<LB> /system/etc/.rild_cfg <LB>
	2	<LB> /system/etc/.dhcpcd <LB>
	3	<LB> -1 <LB>
	4	<LB> sysName <LB>
	5	<LB> 4/system/bin/chmod <LB>
GoldDream	1	<LB> Content-Disposition:form-data; <LB>
	2	<LB> SmsDataType <LB>
	3	<LB> zjphonecall.txt <LB>
	4	<LB> lebar.gicp.net <LB>
	5	<LB> ws_v $< LB >$

per family, we were able to reduce the number of features to 728.[3] These features suggest that unreferenced strings that contain filenames, URLs, and source code are highly informative as to malware families.

Further manual analysis of selected string n-grams also confirmed our initial hypothesis that unreferenced strings present a narrow more focused view of malware behavior and bear interesting information. For example, GoldDream Trojan app uploaded stolen information to a remote server. The URL of this server 'lebar.gicp.net' became visible only through analysis of unreferenced strings (see Table 5). Our analysis also revealed that strings often carry code snippets that contain additional functionality. For example, javascript might be embedded as a string to be executed during the runtime of an app. The use of databases and consequently the presence of sql queries was also prevalent among malware apps.

In preliminary experiments considering this feature selection strategy, the accuracy obtained by using only these 728 features was 96.2%, which is only 3 percentage points lower than using all of the unreferenced strings. However, the false positive rate went up to 13.9%, an increase of 11.9 percentage points. Nevertheless, this feature selection strategy managed to reduce the length of the feature vectors from over 8 million to just 728, while still maintaining relatively high accuracy.

6.3. Discussion

The proposed method of classifying Android malware is rather promising due to several reasons. As opposed to the majority of existing malware classification techniques that rely on a large set of features, the proposed approach requires only 50 features per malware family. This is especially appealing to anti-malware vendors that are constantly forced to keep up with an ever-growing number of signatures for malware detection.

The proposed method exhibits very good filtering capability overall. Our results are especially attractive since it seems to be possible to extract a small set of string features that can effectively identify a suspicious app and further characterize the majority of malware samples within a family. This approach can complement the existing techniques and significantly simplify the triaging stage in automated analysis tools.

[3] Because some tokens were indicative of multiple families, the number is slightly less than the number of families $* 50$.

7. Conclusion

The amount of malware targeting the Android mobile platform is on the rise, as such, the need for an effective automated system to detect and classify malware and the different malware families is crucial. This study presented such a solution, through the use of text strings extracted from the DEX files of an android application. The proposed system of using a linear SVM with line-bounded word-level 3-grams was able to classify malware families with an accuracy of 99.2% while maintaining a false positive rate of just 2.0%.

There are multiple opportunities for future work. First, we will test the feature selection strategy on a dataset of obfuscated apps, as well as attempt to improve the string extraction process by using various natural language processing techniques such as stemming and case folding. Finally, we plan to use the datasets from this study to develop an ensemble classification system which utilizes the string, bytecode, and opcode features.

8. Bibliographical References

Alzahrani, A. J., Stakhanova, N., Gonzalez, H., and Ghorbani, A. (2014). Characterizing evaluation practices of intrusion detection methods for smartphones. *Journal of Cyber Security and Mobility*, 3:89–132.

Apvrille, A. (2012). Guns and smoke to defeat mobile malware. In *Hashdays Conference*.

Apvrille, A. (2013). Playing hide and seek with dalvik executables. In *Hacktivity*.

Arora, A., Garg, S., and Peddoju, S. (2014). Malware detection using network traffic analysis in android based mobile devices. In *Next Generation Mobile Apps, Services and Technologies (NGMAST), 2014 Eighth International Conference on*, pages 66–71.

Arp, D., Spreitzenbarth, M., Hübner, M., Gascon, H., and Rieck, K. (2014). Drebin: Effective and explainable detection of android malware in your pocket. In *Proceedings of the 21th Annual NDSS*.

Crussell, J., Gibler, C., and Chen, H. (2013). Scalable semantics-based detection of similar android applications. In *18th ESORICS*, Egham, U.K.

Enck, W., Gilbert, P., Chun, B.-G., Cox, L. P., Jung, J., McDaniel, P., and Sheth, A. N. (2010). Taintdroid: An information-flow tracking system for realtime privacy monitoring on smartphones. In *Proceedings of the 9th USENIX Conference on OSDI'10*, pages 1–6, Berkeley, CA, USA. USENIX Association.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Gonzalez, H., Stakhanova, N., and Ghorbani, A. (2014). Droidkin: Lightweight detection of android apps similarity. In *Proceedings of International Conference on Security and Privacy in Communication Networks (SecureComm 2014)*.

Gonzalez, H., A.Kadir, A., Alzahrani, A., Stakhanova, N., and Ghorbani, A. A. (2015). Exploring reverse engineering symptoms in android apps. In *European Workshop on Systems Security (EuroSec)*. IEEE.

Gordon, M. I., Kim, D., Perkins, J., Gilham, L., Nguyen, N., and Rinard, M. (2015). Information-flow analysis of Android applications in DroidSafe. In *Proceedings of the 22nd Annual Network and Distributed System Security Symposium (NDSS)*.

Grace, M. C., Zhou, Y., Zhang, Q., Zou, S., and Jiang, X. (2012a). Riskranker: scalable and accurate zero-day android malware detection. In *The 10th MobiSys*, pages 281–294.

Grace, M. C., Zhou, Y., Zhang, Q., Zou, S., and Jiang, X. (2012b). Riskranker: scalable and accurate zero-day android malware detection. In *The 10th MobiSys*, pages 281–294.

Hanna, S., Huang, L., Wu, E., Li, S., Chen, C., and Song, D. (2013). Juxtapp: A scalable system for detecting code reuse among android applications. In *Proceedings of the 9th DIMVA'12*, pages 62–81, Berlin, Heidelberg. Springer-Verlag.

La Polla, M., Martinelli, F., and Sgandurra, D. (2013). A survey on security for mobile devices. *Communications Surveys Tutorials, IEEE*, 15.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pulse Secure Mobile Threat Center. (2015). Mobile threat report. Technical report, Pulse Secure Mobile Threat Center.

Talha, K. A., Alper, D. I., and Aydin, C. (2015). {APK} auditor: Permission-based android malware detection system. *Digital Investigation*, 13:1 – 14.

Tam, K., Khan, S. J., Fattori, A., and Cavallaro, L. (2015). Copperdroid: Automatic reconstruction of android malware behaviors. In *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2014*.

Wolfe, B., Elish, K., and Yao, D. (2014). High precision screening for android malware with dimensionality reduction. In *Machine Learning and Applications (ICMLA), 2014 13th International Conference on*, pages 21–28.

Yan, L. K. and Yin, H. (2012). Droidscope: Seamlessly reconstructing the os and dalvik semantic views for dynamic android malware analysis. In *Proceedings of the 21st USENIX Conference on Security Symposium*, Security'12, pages 29–29, Berkeley, CA, USA. USENIX Association.

Zhang, Y., Yang, M., Xu, B., Yang, Z., Gu, G., Ning, P., Wang, X. S., and Zang, B. (2013). Vetting undesirable behaviors in android apps with permission use analysis. In *Proceedings of the 2013 ACM SIGSAC*, CCS '13, pages 611–622, New York, NY, USA. ACM.

Zhou, Y. and Jiang, X. (2012). Dissecting Android malware: Characterization and evolution. In *IEEE Symposium on Security and Privacy (SP)*, pages 95–109. IEEE.

Zhou, W., Zhou, Y., Jiang, X., and Ning, P. (2012a). Detecting repackaged smartphone applications in third-party android marketplaces. In *Proceedings of the Second ACM CODASPY '12*, pages 317–326, New York, NY, USA. ACM.

Zhou, Y., Wang, Z., Zhou, W., and Jiang, X. (2012b). Hey, you, get off of my market: Detecting malicious apps in official and alternative android markets. In *19th Annual NDSS*.

Zhou, W., Zhou, Y., Grace, M., Jiang, X., and Zou, S. (2013). Fast, scalable detection of "piggybacked" mobile applications. In *Proceedings of the CODASPY '13*, pages 185–196, New York, NY, USA. ACM.

Demystifying Privacy Policies with Language Technologies: Progress and Challenges

Shomir Wilson*, Florian Schaub*, Aswarth Dara*, Sushain K. Cherivirala*, Sebastian Zimmeck†, Mads Schaarup Andersen*, Pedro Giovanni Leon‡, Eduard Hovy*, Norman Sadeh*

*Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213, USA
shomir@cs.cmu.edu, sadeh@cs.cmu.edu

†Columbia University
116th Street & Broadway
New York, NY 10027, USA

‡Stanford University
450 Serra Mall
Stanford, CA 94305, USA

Abstract

Privacy policies written in natural language are the predominant method that operators of websites and online services use to communicate privacy practices to their users. However, these documents are infrequently read by Internet users, due in part to the length and complexity of the text. These factors also inhibit the efforts of regulators to assess privacy practices or to enforce standards. One proposed approach to improving the status quo is to use a combination of methods from crowdsourcing, natural language processing, and machine learning to extract details from privacy policies and present them in an understandable fashion. We sketch out this vision and describe our ongoing work to bring it to fruition. Further, we discuss challenges associated with bridging the gap between the contents of privacy policy text and website users' abilities to understand those policies. These challenges are motivated by the rich interconnectedness of the problems as well as the broader impact of helping Internet users understand their privacy choices. They could also provide a basis for competitions that use the annotated corpus introduced in this paper.

Keywords: Privacy, Internet, annotation, corpus, crowdsourcing.

1. Introduction

Websites' privacy policies are the common (if not nearly pervasive) mechanism by which website operators inform users how their data will be collected, protected, shared, or otherwise processed. However, studies have shown that the average Internet user reads few of the privacy policies of websites they visit (Federal Trade Commission, 2012), would need a substantial amount of time to do so, and even then would have difficulty understanding what those policies mean (McDonald and Cranor, 2008). Moreover, the length and complexity of these documents are a hindrance to policy regulators who are tasked with assessing and enforcing compliance with legal and regulatory requirements. These problems have led to the assessment that the current "notice and choice" model of online privacy is broken (Reidenberg et al., 2014).

These impractical aspects of privacy policies pose a ripe challenge that several efforts have tried to resolve. Some, most notably P3P (W3C, 2006), have relied on voluntary cooperation from website operators to provide formally-specified data on their privacy practices, which are presented to users or applied to browser settings such as cookie management. However, website operators have been hesitant to supply their privacy policies in a machine-readable format, to the extent that P3P has not been widely adopted. Other efforts, such as Terms of Service; Didn't Read (Roy, 2016), have relied chiefly on volunteers to annotate privacy policies. This approach has limits as well, as it relies solely on the attentiveness and dedication of its community to function.

An opportunity exists for language technologies to provide some degree of automation to "bridge the gap" between privacy policies and their audiences. In some ways this is a familiar problem domain; natural language processing on legal text is an active area of research and the legal community has begun to recognize it as well (Mahler, 2015). However, the salience of the problem to the average Internet user's experience makes it unique among applications of natural language processing to legal text. Its timeliness to rising concerns about digital privacy is also a strong motivating factor.

We report on the progress of the Usable Privacy Policy Project, an ongoing effort to use crowdsourcing, natural language processing, and machine learning to extract key information from privacy policy text and present that information to Internet users and policy regulators. We outline our efforts to enlist human annotators—both crowdworkers and expert readers—to gather information from privacy policy text. We have applied a fine-grained policy annota-

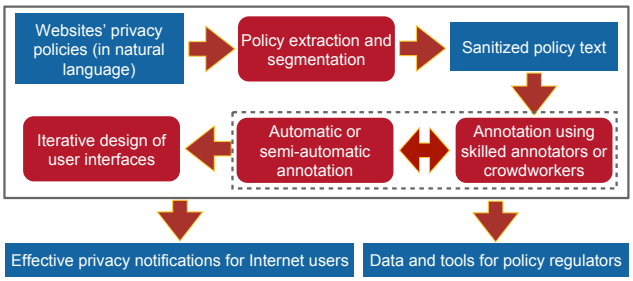

Figure 1: The structure of our approach for processing website privacy policies.

tion procedure to a set of 115 privacy policies (267K words) to label them with a total of 23K data practices, 128K practice attributes, and 103K annotated text spans. This corpus of privacy policies is unprecedented in its size and detail, and we plan to release it to the research community to encourage exploration of this topic. Finally, we describe several research challenges for automating the annotation and analysis of privacy policies. These problems are motivated by the status quo in usable online privacy and the potential broader impacts of improving language technologies.

2. Making Privacy Policies Usable

The Usable Privacy Policy Project[1] (Sadeh et al., 2013) is an NSF-funded Frontier Project with the goals of automatically or semi-automatically extracting key details from privacy policy text, presenting those details to Internet users in formats that respond to their needs, and enabling large-scale privacy policy analysis to inform regulators and key policymakers. We concentrate below on the aspects of the project that are most relevant to language technologies, to motivate the challenges we wish to share with this research community.

Figure 1 illustrates our overall approach to processing privacy policy text. We assume no cooperation on the part of website operators; thus our pipeline begins with website privacy policies as they are generally presented to users, in natural language with some HTML markup. Each privacy policy is downloaded, sanitized to remove extraneous page elements, and then (if appropriate for later stages) divided into segments that roughly correspond to paragraphs. Privacy policies or policy segments are then annotated with information on privacy practices using a combination of human annotators and automatic methods currently under development. We are improving this stage incrementally, by first gathering data from policies with the help of human annotators and then using that data to train models that predict aspects of policy contents. Over time, we expect to increasingly automate the annotation process, limiting the need for human annotators to tasks where human judgement is strictly necessary.

Finally, the results of the annotation process must be presented to Internet users and policy regulators in ways that are responsive to their needs. Our project is developing browser plugins that will show users the privacy practices of websites as they visit them. This is inspired in part by prior work on privacy nutrition labels (Kelley et al., 2009) and privacy profiles (Liu et al., 2014), recognizing the substantial challenges to showing users information about privacy practices. Our team has also developed a data exploration website to showcase the results of some of our annotations. We introduce this site in the next section. The intended outcomes of this project include a transfer of technologies and analysis results to industry, regulators, and policymakers, to ensure a lasting impact of the work.

[1] https://www.usableprivacy.org

3. Collecting Information from Privacy Policies

We have taken a two-pronged approach to human annotation of privacy policy contents, by creating a question-based annotation tool for crowdworkers and a fine-grained annotation tool for expert annotators. We describe both below and explain the future intersection of these approaches.

3.1. Crowdsourcing

Crowdsourcing is a well-recognized method of solving problems that are difficult for computers but easy for humans (Quinn and Bederson, 2011). However, Internet users interpret privacy policies only with great difficulty (Jensen and Potts, 2004; McDonald and Cranor, 2008), potentially ruling out crowdsourcing for this problem. To overcome this limitation, we investigated the accuracy of crowdworkers' answers to questions about privacy policies when multiple workers' answers are aggregated.

Figure 2 shows the interface of our privacy policy annotation tool for crowdworkers. For the task, crowdworkers read the text of a privacy policy, answered nine multiple-choice questions about the privacy practices that it described, and highlighted the text that answered each question. Ten crowdworkers completed this task for each policy, and we experimented with aggregating their answers for each question using a confidence threshold: if the fraction of annotators who chose the most popular answer to a question about a privacy policy was no less than a specified percentage, that answer was designated the crowd's answer. If the most popular answer did not meet the percentage, then it was deemed that the crowd had not produced an answer. Varying the threshold thus permitted tuning for only high agreement answers (high threshold) or for broad coverage (low threshold). Separately, we also experimented with using a logistic regression model to predict which paragraphs were relevant to each question. We highlighted these paragraphs for crowdworkers and measured their impact on workers' accuracy, speed, and confidence.

Our results showed that requiring a high level of agreement produced relatively high accuracy while retaining substantial coverage (Wilson et al., 2016). For example, setting the threshold at 80% agreement resulted in crowd answers for 69% of question-policy pairings and 96% of those answers matched experts' interpretations. We also observed a slight increase in task completion speed with the provision of relevance highlights, without a negative impact on accuracy. Moreover, in an exit survey, annotators who used the interface with highlights expressed greater ease with understanding legal texts than annotators who did

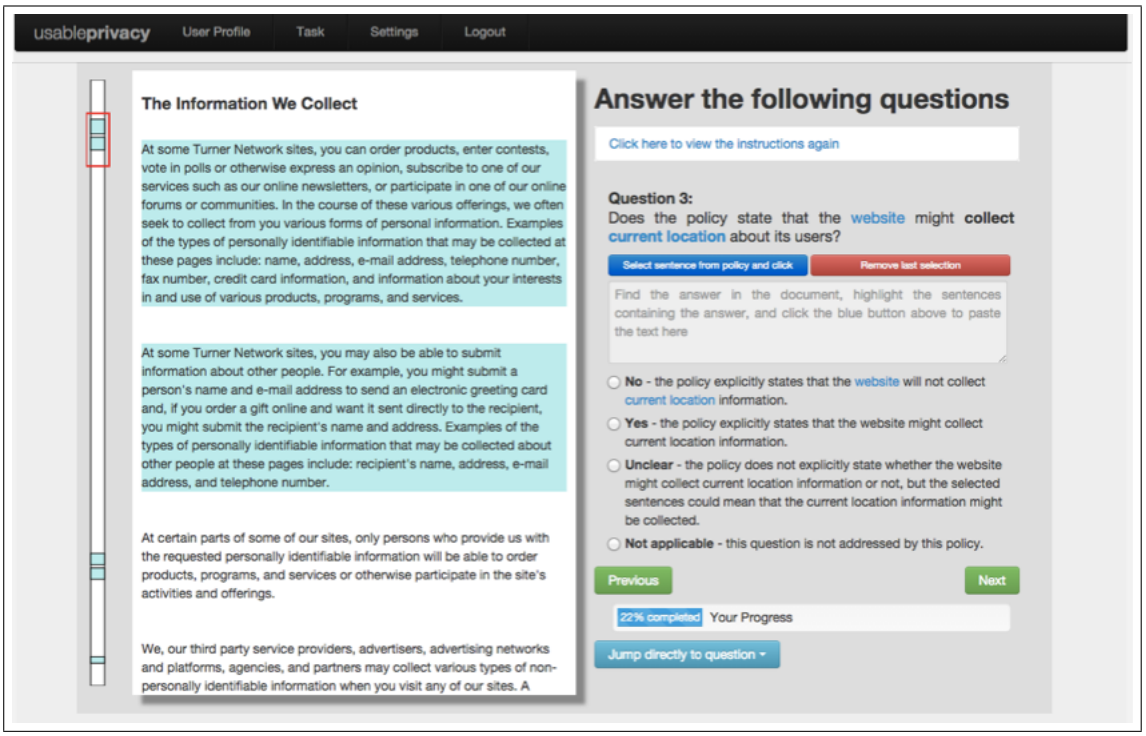

Figure 2: The policy annotation tool for crowdworkers.

not see highlights. This suggests that the task is easier for crowdworkers when highlights are provided, and providing them may reduce fatigue and increase endurance for similar labeling tasks.

3.2. Expert Annotation

The crowdworker annotation tool collected answers to simple questions about privacy policies, and a need remains for annotations that better capture the nuances of privacy practices. Solving this entirely with crowdworkers is highly complex and challenging: even after defining a sufficiently fine-grained annotation scheme, the task is too intricate to give to crowdworkers and interdependencies within it must be discovered to decompose it. Thus, we created a fine-grained annotation tool and enlisted *expert workers* (graduate students in law) to annotate a set of privacy policies. We are planning for an initial release of a corpus of 115 privacy policies plus annotations on the Usable Privacy Policy Project website (URL in Footnote 1). The next steps will be to identify annotation subtasks that can be automated and other subtasks that are suitable for crowdsourcing.

The intent of the fine-grained annotation scheme is to capture the relationship between the data practices (i.e., axioms about how a website user's data is collected, shared, or otherwise applied) intended by a privacy policy and the specific segments of text that express those practices. Each data practice belongs to one of ten categories that broadly express its genus:

1. First Party Collection/Use
2. Third Party Sharing/Collection
3. User Choice/Control
4. User Access, Edit and Deletion
5. Data Retention
6. Data Security
7. Policy Change
8. Do Not Track
9. International and Specific Audiences
10. Other

Each of these ten categories is associated with a set of attributes, and each attribute is associated with a set of potential values. For example, *User Choice/Control* has five attributes: *Choice Type*, *Choice Scope*, *Personal Information Type*, *Purpose*, and *User Choice*. An attribute may be required (the annotator must select a value for the attribute when creating a practice) or optional (since policies are sometimes vague). Crucially, most values are required to be associated with a span of text in the privacy policy.

Figure 3 shows the interface of the fine-grained annotation tool. Expert workers read one policy segment at a time. To create a data practice, the annotator first selects a category and then, for each attribute, specifies a text span (by clicking and dragging) and a value. For example, Figure 3 shows a partly instantiated *User Choice/Control* practice: *Opt-in* is the selected value for *Choice Type*, and it is associated with the highlighted span of text in the policy fragment. A value has been selected for attribute *Purpose* as well, though its text span is not shown. Values have not yet been selected for three attributes, though the lack of an asterisk by attribute *User Type* indicates it is optional.

We have collected annotations for a set of 115 website privacy policies. Websites were selected to represent diverse coverage of sectors (e.g., entertainment, e-commerce,

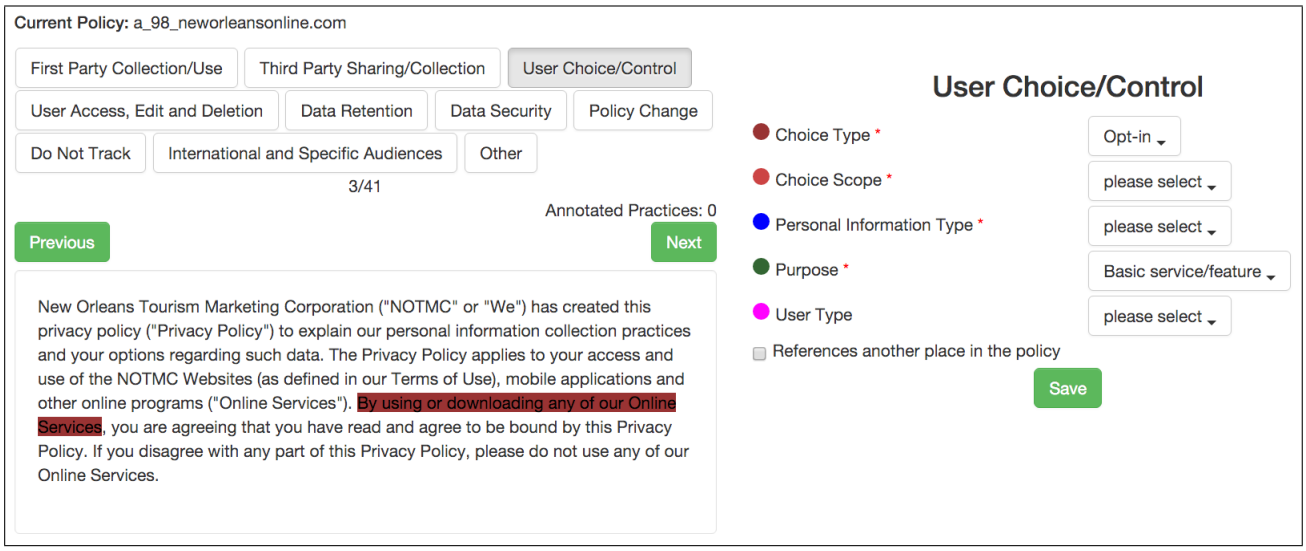

Figure 3: The fine-grained policy annotation tool, for expert annotators.

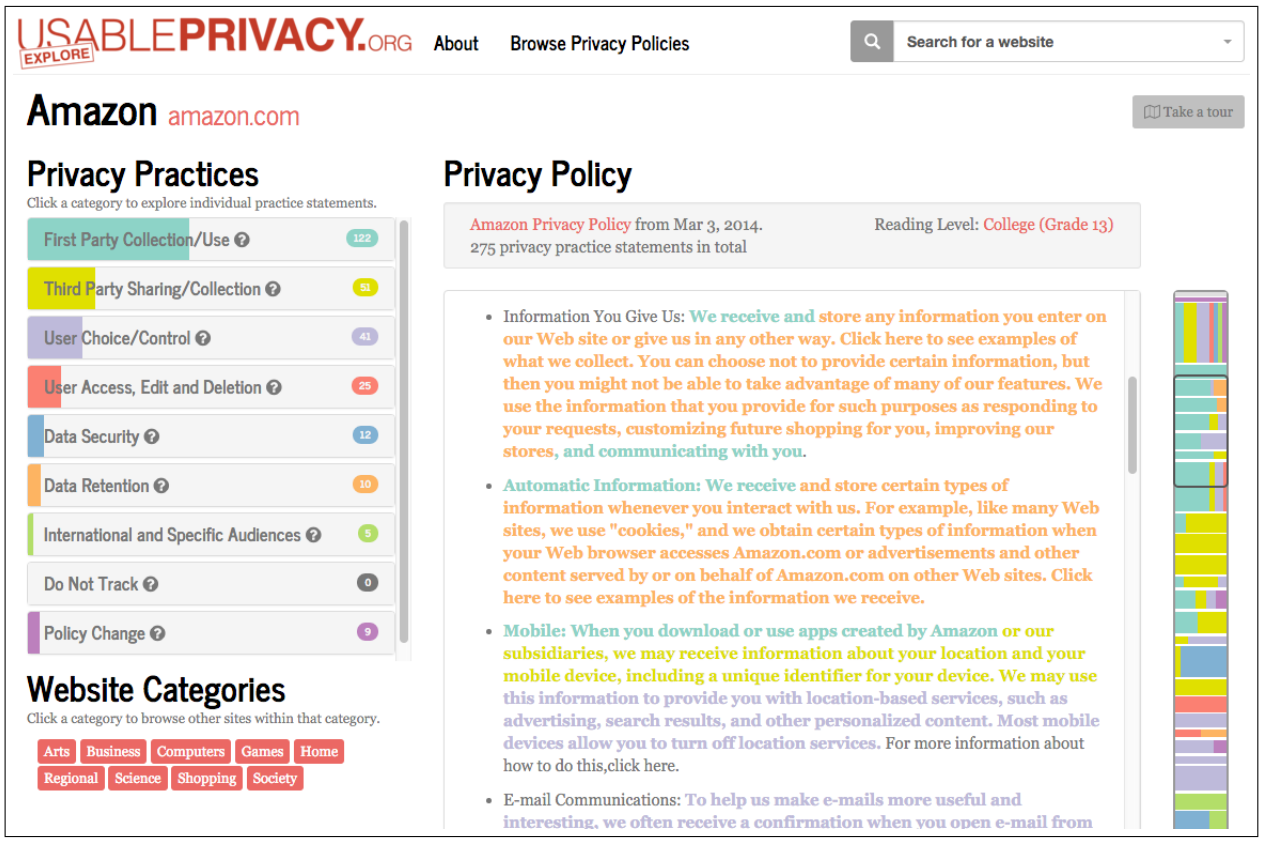

Figure 4: Viewing a sample policy on the data exploration site.

reference, education, etc.) and levels of popularity (as determined by Alexa.com rankings). Each website was annotated by three expert workers who worked independently.

The annotated corpus represents an unprecedented survey of online privacy practices as well as a unique, specialized language resource. In aggregate, the privacy policies consist of 267K words, and the expert workers produced 23K data practices, 128K practice attributes, and 103K annotated text spans associated with those attributes. Notably these counts of annotations represent unconsolidated work—i.e., from three expert workers—and their annotations exhibit some variation. We are exploring several alternatives for identifying and consolidating redundant practices, which is a nontrivial problem because of the complexity of the annotation scheme and thus the variety of modes of divergence.

Finally, Figure 4 is a screenshot of a data exploration

website[2] that we have created to showcase the annotations for the set of 115 website privacy policies. The website allows users to search the collection of privacy policies for specific websites and browse the privacy practices annotated by the expert workers. It also identifies the sectors that a website belongs to, as categorized by DMOZ,[3], and allows the user to compare it to its peers in the same sector.

4. Challenges for the Research Community

The corpus of 115 annotated policies is intended to be a resource for research in natural language processing, usable privacy, and policy analysis. To this end, we challenge these research communities to investigate a family of problems related to the automated analysis of privacy policies. These problems are well-motivated by established topics in natural language processing as well as the difficulties of the "notice and choice" model of online privacy in its current form. Solving them will be progress toward helping Internet users understand how their personal information is used and what choices they can make about that usage. Additionally, policy regulators and creators will have tools to help them monitor compliance with laws and detect trends that require action.

A central challenge of this research direction is the need to annotate privacy policies in a scalable, cost-efficient manner. We have already observed how machine learning can be used to guide human annotators' efforts; for example, the automatically-generated paragraph highlights made the crowdsourcing task (Figure 2) easier for workers. We have also performed preliminary experiments with machine learning to determine that policy segments can be classified automatically for their relevance to practice categories in the fine-grained annotation scheme. These are steps toward a goal of limiting the need for human annotators to small, self-contained tasks that are optimal for crowdsourcing while natural language processing and machine learning take care of the bulk of the analysis. An ambitious (but not completely unreasonable) goal will be to eliminate the need for human annotators altogether. By producing confidence ratings alongside data practice predictions, an automated system could account for its shortcomings by stating which predictions are very likely to be correct and deferring to crowdworkers for predictions that lack firmness.

We propose that the automatic annotation of privacy policies with data practices is a suitable problem for a competitive challenge, and the challenge will advance the state of the art in applicable language technologies. Our corpus of 115 annotated policies contains data that can serve as a gold standard for evaluating solutions. The problem is decomposable into two interrelated subproblems:

- *Prediction of segment relevance to categories*: Given a segment of privacy policy text and one of the ten practice categories, can an automated system predict whether the segment contains a practice belonging to the category? Similarly, can an automated system delimit one or more subsegments of text that correspond with individual data practices?
- *Prediction of values for practice attributes*: Given a segment of policy text and the knowledge that it contains a data practice in a specified category, can an automated system predict the values of the practice's attributes? Similarly, can it identify the text associated with each of those values?

Methods from information retrieval, semantic parsing, coreference resolution, and named entity recognition are particularly relevant to these problems, although a functioning system may use a broad range of methods that we do not immediately anticipate. A sufficiently large space exists for potential solutions that the research community could benefit from an organized challenge as part of a workshop or similar event.

Finally, we are also actively working on or are interested in solving several problems that the corpus will enable us to investigate:

- *Consolidation of annotations from multiple workers*: Under what criteria do a pair of non-identical data practices produced by two annotators refer to the same underlying axiom in the text? Criteria may be observable (i.e., present in the practices' attributes or text spans) or latent (depending on factors such as policy ambiguity or vagueness, which may cause two annotations of an axiom to be divergent without either being in error).
- *Recombination of data practices into a cohesive body of knowledge about a privacy policy*: How do data practices for a privacy policy relate to each other? The answer to this is not contained chiefly in the annotations. For example, two data practices may appear to contradict each other even though they do not, because the reconciliation cannot be represented by the annotation scheme, and thus it is absent from the annotations. Inconsistencies, generalizations, and implications are other examples of potential relationships between data practices. Adding expressiveness to the annotation scheme comes with the tradeoff of greater complexity.
- *Summarization and simplification*: Can the text of a privacy policy be shortened or reworded so that the average Internet user can understand it? A simple test for content equivalence is whether an annotation procedure (by humans or automated methods) produces the same set of data practices for the simplified text and the original text. In practice, Internet users have already demonstrated limited patience with text-based privacy policies, but this problem is nevertheless motivated by the broader goal of making complex texts easier to understand.
- *Optimizing the balance between human and automated methods for privacy policy annotation*: Human annotators and automated annotation both have strengths and weaknesses. The ideal combination in an annotation system will depend on the necessary level of confidence in the annotations and the availability of resources. These resources include human

[2] `https://explore.usableprivacy.org/`

[3] `https://www.dmoz.org/` also used by Alexa.com

annotators, computational power, and training data to create computational models.

- *Identifying sectoral norms and outliers*: Within a sector (e.g., websites for financial services or news), how can we identify typical and atypical practices? A bank website that collects users' health information, for example, would be atypical. When an atypical practice is found, when should it be a cause for concern (or commendation)? Can we recommend websites in a given sector based on an Internet user's expressed privacy preferences?

- *Identifying trends in privacy practices*: The activities that Internet users perform online continue to evolve, and with that evolution the mechanisms for collecting, using, and sharing their data are subject to change. The Internet of Things (IoT) provides a potent example, as sensors collect and share progressively larger amounts of sensitive data. Finding trends in privacy practices will help guide policy regulators to focus their attention on emerging issues.

5. Conclusion

We have described progress toward making privacy policies usable, for the benefit of Internet users, regulators, and policymakers. Additionally, we have presented a rich set of challenges for the research community, engaging efforts in crowdsourcing, natural language processing, and machine learning. The corpus of privacy policy annotations introduced in this paper could also provide a basis for one or more competitions in this area.

6. Acknowledgements

The authors would like to acknowledge the entire Usable Privacy Policy Project team for its dedicated work.

This research has been partially funded by the National Science Foundation under grant agreement CNS-1330596.

7. Bibliographical References

Federal Trade Commission. (2012). Protecting consumer privacy in an era of rapid change: Recommendations for businesses and policymakers.

Jensen, C. and Potts, C. (2004). Privacy policies as decision-making tools: an evaluation of online privacy notices. In *Proc. CHI*. ACM.

Kelley, P. G., Bresee, J., Cranor, L. F., and Reeder, R. W. (2009). A "nutrition label" for privacy. In *Proc. SOUPS*, pages 4:1–4:12, New York, NY, USA. ACM.

Liu, B., Lin, J., and Sadeh, N. (2014). Reconciling mobile app privacy and usability on smartphones: Could user privacy profiles help? In *Proc. WWW*, pages 201–212, New York, NY, USA. ACM.

Mahler, L. (2015). What is NLP and why should lawyers care? *Law Practice Today*.

McDonald, A. M. and Cranor, L. F. (2008). The cost of reading privacy policies. *I/S: J Law & Policy Info. Soc.*, 4(3).

Quinn, A. J. and Bederson, B. B. (2011). Human computation: A survey and taxonomy of a growing field. In *Proc. CHI*, pages 1403–1412, New York, NY, USA. ACM. 00257.

Reidenberg, J. R., Russell, N. C., Callen, A. J., Qasir, S., and Norton, T. B. (2014). Privacy harms and the effectiveness of the notice and choice framework. SSRN Scholarly Paper.

Roy, H. (2016). Terms of Service; Didn't Read. https://tosdr.org/. Accessed 2016-02-15.

Sadeh, N., Acquisti, A., Breaux, T. D., Cranor, L. F., Mcdonald, A. M., Reidenberg, J. R., Smith, N. A., Liu, F., Russell, N. C., Schaub, F., and Wilson, S. (2013). The usable privacy policy project: Combining crowdsourcing, machine learning and natural language processing to semi-automatically answer those privacy questions users care about. Technical Report CMU-ISR-13-119, Carnegie Mellon University.

W3C. (2006). The Platform for Privacy Preferences 1.1 (P3P1.1) Specification. `https://www.w3.org/TR/P3P11/`. Last accessed: Mar. 31, 2016.

Wilson, S., Schaub, F., Ramanath, R., Sadeh, N., Liu, F., Smith, N. A., and Liu, F. (2016). Crowdsourcing annotations for websites' privacy policies: Can it really work? In *Proc. WWW*, Montreal, Canada. ACM.