The 2016 AAAI Fall Symposium Series:
Privacy and Language Technologies
Technical Report FS-16-04

Lexical Similarity of Information Type Hypernyms, Meronyms and Synonyms in Privacy Policies

Mitra Bokaei Hosseini,[1] Sudarshan Wadkar,[2] Travis D. Breaux,[2] Jianwei Niu[1]
University of Texas at San Antonio, San Antonio, Texas[1]
Carnegie Mellon University, Pittsburgh, Pennsylvania[2]
mitra.bokaeihosseini@utsa.edu, {swadkar,breaux}@cmu.edu, jianwei.niu@utsa.edu

Abstract

Privacy policies are used to communicate company data practices to consumers and must be accurate and comprehensive. Each policy author is free to use their own nomenclature when describing data practices, which leads to different ways in which similar information types are described across policies. A formal ontology can help policy authors, users and regulators consistently check how data practice descriptions relate to other interpretations of information types. In this paper, we describe an empirical method for manually constructing an information type ontology from privacy policies. The method consists of seven heuristics that explain how to infer hypernym, meronym and synonym relationships from information type phrases, which we discovered using grounded analysis of five privacy policies. The method was evaluated on 50 mobile privacy policies which produced an ontology consisting of 355 unique information type names. Based on the manual results, we describe an automated technique consisting of 14 reusable semantic rules to extract hypernymy, meronymy, and synonymy relations from information type phrases. The technique was evaluated on the manually constructed ontology to yield .95 precision and .51 recall.

Introduction

Mobile and web applications (apps) are increasingly popular due to the convenient services they provide in different domains of interest. According to the PEW Research Center, 64% of Americans own a smart phone (Smith, 2015). They found that smart phone users typically check health-related information online (62% of Americans), conduct online banking (54%), and look for job-related information (63%). To fulfill user needs and business requirements, these apps collect different categories of personal information, such as friends' phone numbers, photos and real-time location. Regulators require apps to provide users with a legal privacy notice, also called a privacy policy, which can be accessed by users before installing the app. For example, the California Attorney General's office recommends that privacy policies list what kinds of personally identifiable data are collected, how it is used, and with whom it is shared (Harris, 2013). However, data practices are commonly described in privacy polices using hypernymy (Bhatia, Evans, Wadkar, & Breaux, 2016), which occurs when a more abstract information type is used instead of a more specific information type. Hypernymy allows for multiple interpretations, which can lead to ambiguity in the perception of what personal information is collected, used or shared. To address this problem, we applied content analysis, which is a qualitative research method for annotating text to identify words and phrases that embody the meaning of special codes (Saldaña, 2015), and grounded theory (Corbin & Strauss, 2014) to discover heuristics for manually classifying information types into a formal ontology. We evaluated these heuristics in a second study of 50 mobile app privacy policies. Furthermore, we developed an automated technique to replace the manual method and discover prospective hypernyms, meronyms and synonyms. This technique consists of 14 reusable semantic rules that characterize how to infer these relationships directly from phrases.

This paper is organized as follows: first, we discuss background and related work; then, we introduce our content analysis method and results, including the seven heuristics; then, we describe our automated technique for discovering hypernym, meronym and synonym prospects, before presenting results of evaluating this technique against the manually constructed ontology. We conclude with future work.

Important Terminology

The following terms are used throughout this paper:

- *Hypernym* – a noun phrase, also called a superordinate term, that is more generic than another noun phrase, called the *hyponym* or subordinate term.
- *Meronym* – a noun phrase that represents a part of a whole, which is also a noun phrase and called a *holonym*.

- *Synonym* – a noun phrase that has a similar meaning to another noun phrase.
- *Lexicon* – a collection of phrases or concept names that may be used in an ontology.
- *Ontology* – a collection of concept names and relationships between these concepts, including hypernym, meronym and synonym relationships.

Background and Related Work

In software engineering, privacy policies are critical requirements documents that are used by various stakeholders to communicate about data practices (Anton & Earp, 2004). Due to different needs and background context, there can be disparate viewpoints and assumptions regarding what is essentially the same subject matter (Uschold & Gruninger, 1996). Stakeholders can use different words for the same domain, which reduces shared understanding of the subject. This confusion can lead to a misalignment among what designers intend, what policies say, and what regulators expect (Breaux & Baumer, 2010).

In requirements engineering, Potts and Newstetter identify two approaches to codifying knowledge: naïve positivism, and naturalistic inquiry (Potts & Newstetter, 1997). Positivism refers to the world with a set of stable and knowable phenomena, often with formal models. Naturalistic inquiry (NI) refers constructivist views of knowledge that differs across multiple participant observations. The research in this paper attempts to balance among these two viewpoints by recognizing that information types are potentially unstable and intuitive concepts. Our approach initially permits different conceptual interpretations, before reducing terminological confusion to reach a shared understanding. For example, terminological confusion arises in the mobile privacy policy phrase "network information," which is an abstract information type that is occasionally used by policy writers in privacy policies. This term can mean any data sent over a network as well as network configuration information, or it might only mean IP address.

Formal ontologies allow us to achieve shared meaning by relating concepts to each other using logical relationships with hypernymy, meronymy and synonymy, among others (Martin & Jurafsky, 2000). An ontology can be used for disambiguation, when policies contain head words that are hypernyms, e.g., the word "address" in a policy can mean either an "IP address" or "e-mail address". We now review prior research on ontology in privacy, before discussing existing methods to construct ontologies.

Ontology in Security and Privacy Policy

In prior work, ontologies have been developed for privacy. Heker et al. developed a privacy ontology for e-commerce transactions (Hecker, Dillon, & Chang, 2008). The lexicon they used to implement the ontology includes information about privacy mechanisms and privacy principles from legislative documents, such as European Parliament Directive 95/46/EC. The ontology includes general entities for e-commerce transactions, such as authentication, authorization, and identities.

In the domain of security policies, Bradshaw et al. presented KAoS, a policy service framework which includes a user interface for presenting a natural language policy specifications, an ontology management component for expressing and reasoning over Description Logic (DL) ontologies, and a policy monitoring and enforcement layer that compiles the specifications into machine readable policies (Bradshaw, et al., 2003). Using this framework, intelligent agents continually adjust their behavior with specifications. Kagal et al. constructed an ontology to enforce access control policies in a web services model (Kagal, et al., 2004). This ontology is expressed in RDF and OWL-Lite and describes user and web service specifications about the information users agreed to share with web services. Syed et al. developed an ontology in OWL DL which provides a common understanding of cybersecurity domain and unifies most commonly used cybersecurity standards (Syed, Padia, Finin, Mathews, & Joshi, 2016). This ontology which is called Unified Cybersecurity Ontology (UCO) also map some existing publicly available cybersecurity ontologies to promote ontology sharing, integration and reuse.

Breaux et al. utilized an ontology to find conflicts between the privacy policies regarding data collection, usage, and transfer requirements (Breaux, Hibshi, & Rao, 2014). An ontology was used to infer data flow traces across separate policies in multi-tier applications (Breaux, Smullen, Hibshi, 2015). These ontologies include simple hierarchies for actors, information types and purpose expressed in DL.

To our knowledge, our work is the first privacy-related ontology that formally conceptualizes personally identifiable information types and their relationships from 50 mobile app privacy policies. This paper describes the empirically validated, bootstrap method used to create the ontology, as well as new techniques for automating the method. Moreover, the initial version of the manually constructed ontology has been used to find conflicts between mobile app code-level method calls and privacy policies (Slavin et al., 2016).

Constructing an Ontology

According to Uschold and Gruninger, there is no standard method to build an ontology (Uschold & Gruninger, 1996). However, a general approach includes: identifying the purpose and scope for the ontology; identifying key concepts that lead to a lexicon; identifying the relationships between concepts in the lexicon; and formalizing those relationships.

A lexicon consists of terminology in a domain, whereas ontologies organize terminology by semantic relationships, including *hypernyms*, which describe super- and sub-ordinate conceptual relationships, *meronyms*, which describe part-whole relationships, and *synonyms*, which describe different words with similar or equivalent meanings (Huang, 2010). Lexicons can be constructed using content analysis of source text, which yields an annotated corpora. Breaux and Schaub empirically evaluated crowdsourcing as a means to create corpora from annotated privacy policies (Breaux & Schaub, 2014). Wilson et al. described the creation of a privacy policy corpus from 115 privacy policies using crowdsourcing (Wilson, et al., 2016).

Marti Hearst proposed six lexico-syntactic patterns to automatically identify hypernymy in natural language text using noun phrases and regular expressions (Hearst, 1992). Example patterns are "such as", "including", and "especially." After identifying these patterns in a sentence, the associated noun phrases are extracted using part of speech tags. The noun phrases are then verified by comparison with an early version of WordNet, which is a popular lexical database that contains hyponyms (Miller, 1995).

Snow et al. presented a machine learning (ML) approach for learning hypernymy relationships in text which also relies on lexico-syntactic patterns (Snow, Jurafsky, & Ng, 2004). The ML features are derived from hypernym-hyponym pairs in WordNet, which are then found in parsed sentences of newswire corpus. A resulting syntactic dependency path is used to describe each pair. This ML approach relies on explicit expressions of hypernymy in text, whereas we discovered a rule-based approach to identify hypernyms based on shared words among information type phrases.

By comparison to Hearst and Snow et al., Bhatia et al. applied an extended set of Hearst-related patterns to 15 privacy policies and found that this approach yields hypernyms for only 23% of the lexicon (Bhatia, Evans, Wadkar, & Breaux, 2016). This means the remaining 77% of the lexicon must be manually analyzed to construct an ontology.

Ontology Construction Overview

The ontology construction method (see Figure 1) consists of 6 steps: steps 1-3 are part of a crowdsourced content analysis task based on Breaux and Schaub (2014); and step 4 employs an entity extractor developed by Bhatia and Breaux (2015), to yield a lexicon (artifact A). In this paper, we extend that prior work with a novel set of heuristics for manually classifying information types from a lexicon into an ontology (step 5), including a technique to automatically identify prospective hypernym, meronym, and synonym relationships (step 6).

Acquiring the Mobile Privacy Policy Lexicon

The mobile privacy policy lexicon (artifact A in Figure 1) was constructed using a combination of crowdsourcing, content analysis and natural language processing (NLP). We first selected the top 20 English policies across each of 69 categories in Google Play[1], from which we selected 50 mobile app privacy policies. In step 2, the 50 policies were segmented into ~120 word paragraphs using the method described by Breaux & Schaub (2014); this yielded 5,932 crowd worker tasks with an average 98 words per task.

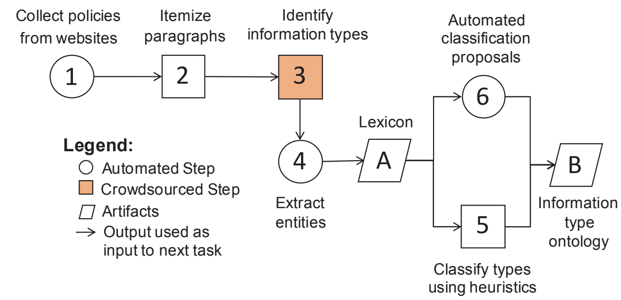

Figure 1. Overview of ontology construction method

In the annotator task (see Figure 2, for example), the annotators identified phrases that correspond to one of two concepts:

- *Platform Information:* any information that the app or another party accesses through the mobile platform which is not unique to the app.
- *Other Information:* any other information the app or another party collects, uses, shares or retains.

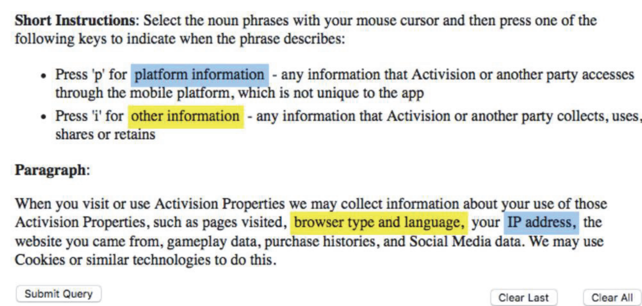

Figure 2. Example crowdsourced annotation task

These two concepts comprise the entire coding frame in content analysis. Example phrases that match the *platform information* code include "IP address," "contact folder," "location information," "email address," and "unique device identifier." The annotators were allowed to work on the tasks at their own pace, taking breaks when they experienced fatigue or boredom. The *other information* code was used to ensure that annotators remained vigilant, particularly when platform information types were sparse in the policy text. To construct the lexicon, we selected only platform information when annotations two or more annotators agreed on the annotation. This number follows the empirical anal-

[1] https://play.google.com

ysis of Breaux & Schaub (2014), which shows high precision and recall for two or more annotators on the same task. Next, we applied an entity extractor (Bhatia & Breaux, 2015) to the remaining annotations to itemize the platform information types into unique entities to be included in the privacy policy lexicon.

Six annotators, including the first, third, and fourth authors performed the annotations. The cumulative time to annotate all tasks was 19.9 hours across all six annotators, which yielded a total 720 unique annotations in which two or more annotators agreed on the annotation.

In the next step, the annotations were analyzed by an entity extractor developed by Bhatia and Breaux (2015). The extractor yields unique information type names from annotations by identifying type boundaries from annotated word lists and incomplete annotations. Based on the 50 policies, the extractor yielded 355 unique information type names.

Manual Ontology Construction

We now describe our bootstrap method for constructing a formal ontology from an information type lexicon. This includes our choice of formalism, the tools used to express the ontology, and the construction method.

Description Logic (DL) ontologies enable automated reasoning, including the ability to infer which concepts subsume or are equivalent to other concepts in the ontology. We chose the DL family $\mathcal{AL}$, which is PSPACE-complete for concept satisfiability and concept subsumption. In this paper, reasoning in DL begins with a TBox T that contains a collection of concepts and axioms based on an interpretation $\mathfrak{I}$ that consists of a nonempty set $\Delta^{\mathfrak{I}}$, called the *domain of interpretation*. The interpretation function $.^{\mathfrak{I}}$ maps concepts to subsets of $\Delta^{\mathfrak{I}}$: every atomic concept C is assigned a subset $C^{\mathfrak{I}} \subseteq \Delta^{\mathfrak{I}}$, the top concept $\top$ has the interpretation $\top^{\mathfrak{I}} = \Delta^{\mathfrak{I}}$.

This $\mathcal{AL}$ family includes operators for concept union and intersection, and axioms for subsumption, and equivalence with respect to the TBox. Subsumption is used to describe individuals using generalities, and we say a concept C is subsumed by a concept D, written $T \vDash C \sqsubseteq D$, if $C^{\mathfrak{I}} \subseteq D^{\mathfrak{I}}$ for all interpretations $\mathfrak{I}$ that satisfy the TBox T. The concept C is equivalent to a concept D, written $T \vDash C \equiv D$, if $C^{\mathfrak{I}} \subseteq D^{\mathfrak{I}}$ for all interpretations $\mathfrak{I}$ that satisfy the TBox T.

We chose DL, because we only aim to identify which lexicon phrases share interpretations. For a given information type in a privacy policy, we query a TBox to identify related types. In the future, we propose to extend the method to discover the exact relationships among types. For example, parts of wholes are formally interpreted using subsumption, because subsumption is not strictly limited to hyponyms. Our approach can be extended to separately reason across hypernyms and meronyms using the DL family SI that includes role transitivity, which is PSPACE-complete for TBox satisfiability (Horrocks, Sattler, & Tobies, 1999). In this family, a meryonym role can distinguish parts of concepts that are not co-transitive with a hypernym role to yield unintended answers, such as a mobile device "sensor" is a kind "mobile device." We express the DL ontology using the Web Ontology Language[2] (OWL) version 2 DL and the Protégé[3] tool version 4.3, which is a graphical tool for manipulating the ontology. OWL has a decentralized philosophy which allows incremental building of knowledge, and its sharing and reuse (Syed, Padia, Finin, Mathews, & Joshi, 2016).

The bootstrap method begins with a "flat" ontology, which is automatically generated to contain concepts names for each information type name. In the flat ontology, every concept name C is only a direct subclass of the top concept, $C \sqsubseteq \top$. Next, two analysts define subsumption and equivalence axioms for concept pairs using Protégé by making paired comparisons among the concepts in the ontology. This method is subject to cognitive bias, including the proximity of concepts to each other in the alphabetical list, and to the recency with which the analysts encountered concepts for comparison (Postman & Phillips, 1965).

The bootstrap method was piloted by the second and third authors on five privacy policies. The pilot study resulted in a set of seven heuristics that form a grounded theory and that explain why two concepts share an axiom in the ontology. For a pair of concepts C_1, C_2, the analysts assign an axiom with respect to a TBox T and one heuristic as follows:

- Hypernym (H): $C_1 \sqsubseteq C_2$, when concept C_2 is a general category of C_1, e.g., "device" is subsumed by "technology".
- Meronym (M): $C_1 \sqsubseteq C_2$, when C_1 is a part of C_2, e.g., "internet protocol address" is subsumed by "internet protocol".
- Attributes (A): $C_1\_C_2 \sqsubseteq C_2$ and $C_1\_C_2 \sqsubseteq C_1\_information$, when the $C_1\_C_2$ phrase contains the C_1 phrase as an attribute or modifier of C_2 phrase, e.g., "unique device identifier" is subsumed by "unique information" and "device identifier".
- Plural (P): $C_1 \equiv C_2$, when the C_1 phrase is a plural form of the C_2 phrase, e.g., "MAC addresses" is the plural form of "MAC address".
- Synonym (S): $C_1 \equiv C_2$, when C_1 is a synonym of C_2, e.g., "geo-location" is equivalent to "geographic location".
- Technology (T): $C_1 \equiv C_1\_information$, when C_1 is a technology, e.g., "device" is equivalent to "device information".

[2] https://www.w3.org/TR/owl-guide/

[3] http://protege.stanford.edu/

- Event (E): $C_1 \equiv C_1\_information$, when C_1 is an event, e.g., “usage” is equivalent to “usage information”.

The first and third author conducted a *4-step heuristic evaluation* of the seven heuristics using the lexicon produced by the 50 mobile app privacy policies as follows: (1) two analysts separately apply the bootstrap method on a copy of the lexicon; (2) for each ontology, an algorithm extracts each expressed and inferred axiom between two concepts using the HermiT[4] reasoner; (3) each relationship assigned to a concept pair appears in one column per analyst (in Figure 3, see Analyst1, where ‘Super’ means the LHS is a superclass of the RHS, ‘Sub’ means subclass of, ‘Equiv’ means equivalence, and ‘None’ means no relationship); and (4) each analyst then separately reviews their assigned axiom type, choose the heuristic to match the assignment, and decides whether to retain or change their axiom type.

In Figure 3, the left-hand side (LHS) concept is compared to the right-hand side (RHS) concept by Analyst1 and Analyst2, whose axiom types appear in their respective column, e.g., Analyst1 assigned “Equiv” to “web pages” and “web sites” and the heuristic “S” to indicate these two concepts are synonyms, whereas Analyst2 assigned “Sub” and heuristic “M” to indicate “web pages” is a part of “web sites.”

LHS Concept	RHS Concept	Heuristic	Analyst1	Analyst2
web pages	web sites	S/M	Equiv	Sub
ads clicked	usage info	H	Sub	Sub
computer	platform	H	Super	Super
log information	system activity	M	None	Super
device type	mobile device type	A	Super	None
tablet	tablet information	T	None	Equiv

Figure 3. Example table comparing concept relationships

Before and after step 3, we compute the Fleiss’ Kappa statistic, which is a chance-corrected, inter-rater reliability statistic (Fleiss, 1971). Increases in this statistic indicate improvement in agreement above chance.

Automated lexeme variant inference

Information types are frequently variants of a common lexeme, for example, “mobile device” is a variant of “device,” called the head word. The relationship among variants can be explained by the heuristics, and we designed a method to automatically infer variants based on semantic rules. Figure 4 shows an example phrase, “mobile device IP address” that is decomposed into the atomic phrases: “mobile,” “device,” and “IP address,” based on a 1-level typology.

The typology links atomic phrases to whether they are one of five kinds: *attributes*, which describe the quality of a thing, such as “mobile” and “personal;” *things*, which is a concept that has logical boundaries and which can be composed of other things; *events*, which describe action performances, such as “usage,” “viewing,” and “clicks;” *agents,* which describe actors who perform actions or possess things; and the special type α which includes “information,” “data,” “details,” and any other synonyms of “information.”

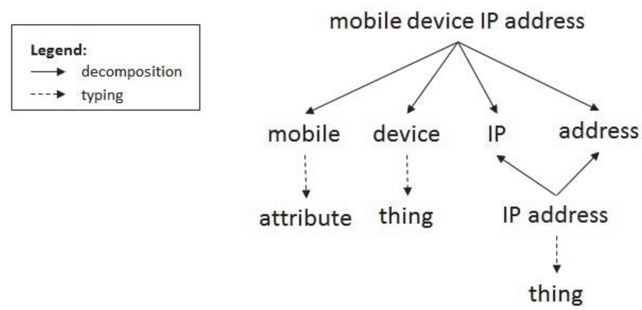

Figure 4. Example Lexicon phrase, grouped and typed

The typed, atomic phrases are recomposed using rules to infer variants. Rules consist of a type pattern, a heuristic, and an inferred ontological relationship. The following rules R1-R13 are expressed using the types: attribute (A), thing (T), event (E), agent (G), and α. Subscripts indicate the order of same-typed phrases in asymmetric ontological relations:

R1. A_T implies $A\_T \sqsubseteq (\text{A\_information} \sqcup T)$ (heuristic A), e.g., “mobile device identifier” is a kind of “mobile information” and “device identifier”.

R2. $T_1\_T_2$ implies $T_1\_T_2 \sqsubseteq (T_1 \sqcup T_2)$ (heuristic M and heuristic H), e.g., “internet protocol address” is a part of “internet protocol” and a kind of “address”.

R3. T implies $T \equiv T\_information$ (heuristic T), e.g., “device” is a synonym of “device information”.

R4. A_T_α implies $A\_T\_\alpha \sqsubseteq (A\_\alpha \sqcup T\_\alpha)$ (heuristic A), e.g., “mobile device information” is a kind of “mobile information” and “device information”.

R5. $T_1\_T_2\_\alpha$ implies $T_1\_T_2\_\alpha \sqsubseteq (T_1\_\alpha \sqcup T_2\_\alpha)$ (heuristic H), e.g., “device log information” is a kind of “device information” and “log information”.

R6. E implies $E \equiv E\_information$ (heuristic E), e.g., “usage” is a synonym of “usage information”.

R7. E_T implies that $E\_T \sqsubseteq (E \sqcup T)$, e.g., “click count” is part of “click” and a kind of “count”.

R8. T_E implies that $T\_E \sqsubseteq (T \sqcup E\_tense\_corrected)$, only if E is tagged as a verb. Otherwise, $T\_E \sqsubseteq (T \sqcup E)$, e.g., “pages viewed” is a kind of “pages” and “views,” which is corrected to present simple, third-person tense.

R9. $T_1\_E\_T_2$ implies that $T_1\_E\_T_2 \sqsubseteq (T_1\_E \sqcup E\_T_2 \sqcup T_1\_information \sqcup T_2\_information)$, e.g., “website activity date” is a part of “website activity” and a kind of “activity date”.

[4] http://www.hermit-reasoner.com/

R10. T_E_α implies that $T\_E\_\alpha \sqsubseteq (T \sqcup T\_\alpha \sqcup E\_\alpha)$, e.g, "language modeling data" is a part of "language" and a kind of "language data" and "modeling data".

R11. A_G_α implies that $A\_G\_\alpha \sqsubseteq (A\_\alpha \sqcup G\_\alpha)$, e.g. "aggregated user data" is a kind of "aggregated data" and "user data".

R12. $A_1\_A_2\_\alpha$ implies that $A_1\_A_2\_\alpha \sqsubseteq (A_1\_\alpha \sqcup A_2\_\alpha)$, e.g., "anonymous demographic information" is a kind of "anonymous information" and "demographic information".

R13. G_T implies that $G\_T \sqsubseteq (G\_information \sqcup T)$, e.g., "user content" is a kind of "user information" and "content".

The above rules were discovered by the first and third author who classified the 355 lexicon phrases using the typology as a second-cycle coding frame (Saldaña, 2015). The automated technique applies the rules to phrases and yields inferred relations for evaluation in three steps: (1) a phrase from the lexicon is decomposed and typed, once, as shown in Figure 4; (2) the semantic rules are matched to the typed phrases to infer new candidate phrases and relations; (3) for each inferred phrase, we repeat step 2 with the inferred phrase. The technique terminates when no rules match a given input phrase. For example, in Figure 4, we perform step (2) by applying the rule R1 to infer that "mobile device IP address" is a kind of "device IP address" based on heuristic A. However, the phrase "device IP address" is not in the lexicon, i.e., it is potentially a *tacit concept name*. Thus, we re-apply the rules and rule R2 matches this phrase's typing to infer that "IP address" is part of "device," which are two *explicit concept names* in the lexicon. Thus, we accept both inferences for further evaluation.

Heuristic P establishes equivalence relations between plural and singular noun phrases as described by R14:

R14. For any plural form of a phrase, this phrase is equivalent to its singular form, e.g., "access devices" is equivalent to "access device."

The R14 relies on part-of-speech (POS) tags to identify plural nouns. A mapping is maintain between plural forms tagged "NNS" and singular forms tagged "NN," which can be discovered in the lexicon based on suffixes –ies, -es, etc. After each word in a phrase is POS-tagged, the words with "NNS" tags in each lexicon phrase are reduced to singular form using the mapping. Finally, an equivalence relation is expressed between the original plural phrase and the inferred singular form. The resulting equivalence can be between explicit and tacit concept names. For example, R14 yields "unique application number" from the phrase "unique application numbers," which are deemed equivalent.

The automated technique yields relation prospects that we evaluate using the manually constructed, ground truth (GT) ontology. We first compare the axioms in the prospective ontology with the GT ontology to measure precision and recall of expressed subclass and equivalence relations. Next, we use the HermiT Reasoner to compute the entailment of the prospective ontology and GT ontology to measure precision and recall of inferred axioms. The second evaluation shows how transitivity and equivalence explains any changes or improvements in precision and recall.

Evaluations and Results

We now describe our results from the manual ontology construction and automated lexeme variant inference.

Manual ontology construction evaluation

The ontology was constructed using the bootstrap method and evaluated in two iterations (see Figure 5): Round 1 covered 25/50 policies to yield 235 concept names and 573 axioms from the 4-step heuristic evaluation, and Round 2 began with the result of Round 1 and added the concepts from the remaining 25 policies to yield a total 368 concept names and 849 axioms. The resulting ontology produced 13 new concepts that were not found in the lexicon, because the analysts added tacit concepts to fit lexicon phrases into existing subsumption hierarchies. Figure 5 presents the results of the number of "Super," "Sub," and "Equiv" axioms and "None" identified after the bootstrap method.

Iteration	Analyst	Super	Sub	Equiv	None
Round 1	1	151	203	77	142
	2	157	172	78	166
Round 2	1	304	343	142	60
	2	313	352	151	33

Figure 5. Number of ontological relations identified by each analyst during each round

	Round 1 (c=235, a=573)		Round 2 (c=368, a=849)	
	Initial	Reconciled	Initial	Reconciled
Agreed	252	543	743	808
Disagreed	321	30	106	12
Consensus	43.9%	94.8%	87.5%	98.4%
Kappa	0.233	0.979	0.813	0.977

Figure 6. Number of agreements, disagreements and Kappa for c concepts and an axioms per round

Figure 6 presents agreements, disagreements, the consensus (ratio of agreements over total axioms compared), and Kappa for the bootstrap method without reconciliation, called *Initial*, and after reconciliation, called *Reconciled*. The round 1 ontology began with 235 concepts and 573 relations from both analysts. The round 2 ontology extended the reconciled round 1 ontology with 132 new concepts.

Figure 7 presents the number of heuristics by type that are assigned to relations by two analysts: (H)ypernym, (M)eronym, (A)ttribute, (P)lural, (S)ynonym, (T)echnology, and (E)vent. Multiple heuristics may apply to some phrases, such as comparing "device information" to "mobile device IP address," which can be compared using the H, A, and M heuristics depending on which order the analyst applies the heuristics. The automated approach, which we now discuss, resolves this ambiguity by decomposing each heuristic into separate rules and applying all relevant rules to each phrase.

Heuristic	**Round 1**		**Round 2**	
Analyst	**1**	**2**	**1**	**2**
Hypernym	349	354	248	357
Meronym	38	38	100	56
Attribute	29	36	108	39
Plural	13	13	12	13
Synonym	55	55	57	57
Technology	10	10	17	15
Event	0	0	4	3

Figure 7. Number of heuristics applied by type

Automated lexeme variant inference evaluation

The 14 rules to infer new information type variants were applied to the 355-phrase lexicon. Figure 8 shows the number of phrases that match each rule in each level (L#) of recursion, e.g., R2 matched 88 phrases and recursively matched 24 variants after other rules were applied. The technique yields 865 phrases after typing and decomposition, which consists of 355 *explicit concept names* from the original lexicon, and 510 potential *tacit concept names*, and it yielded 1607 total axioms. Rule R3 was most frequently used, including recursively after "things" were separated from "attributes" and other "things."

Rule	**Pattern**	**No. phrases, matched**			
		L1	L2	L3	L4
R1	A-T	52	-	-	-
R2	T_1-T_2	88	24	4	-
R3	T	158	201	42	4
R4	A-T-α	8	-	-	-
R5	T_1-T_2-α	28	7	-	-
R6	E	7	27	18	-
R7	E-T	23	11	-	-
R8	T-E	10	9	-	-
R9	T_1-E-T_2	11	-	-	-
R10	T-E-α	8	3	-	-
R11	A-G-α	1	-	-	-
R12	A_1-A_2-α	17	1	-	-
R13	G-T	4	-	-	-

Figure 8. Number of phrases matched per rule, including matches per level (L#) of recursive rule applications

We compute precision and recall using the GT ontology, as follows: an axiom is counted as a true positive (TP), only if it appears in the GT ontology with reasoning. Otherwise, it is counted as false positive (FP). Figure 9 shows the precision (Prec.) and recall (Recall) for the automated technique: *expressed axioms* are generated by the technique and included in the prospective ontology; *entailed axioms* are entailed by the HermiT Reasoner. The evaluation is over the subset of 711 GT ontology axioms wherein concept names share one or more words in the lexicon. Despite this limitation, the technique produces few FPs that are all explained as analyst omissions during manual construction.

	Subclasses		**Equivalence**	
Axioms	Prec.	Recall	Prec.	Recall
Expressed	0.446	0.446	0.765	0.302
Entailed	0.956	0.653	0.944	0.361

Figure 9. Evaluation of Subsumption and Equivalence Relations

Overall, the automated technique correctly identifies 41% or 293/711 of hypernyms, meronyms and synonyms in the 355-concept GT ontology. We observed that 59% or 118/199 of false negatives (FNs) are between concept names that require an additional ontology to reason about similarity, exceeding the limits of our typology. For example, to discover that "mobile phone" is a kind of "mobile device," we need to know that a "phone" is a kind of "device." Adding this ontology could potentially improve recall to 0.891 and 0.596 for sub- and equivalent classes, respectively.

We observed that 18/60 FNs of equivalence relations require further domain understanding, e.g., "postal code" is equivalent to "zip code," or in case of acronyms, "internet protocol address" is equivalent to "IP address." Finally, we discovered that 24/199 total FNs were due to GT ontology errors from inconsistencies with the automated technique, e.g., an analyst's equivalence axiom was identified by the technique as subclass axiom, and all 14/14 FPs are axioms missed by the analysts.

Discussion and Future Work

We now discuss our results and the impact of our work. We present an automated technique that we evaluated on a 355 phrase lexicon acquired from 50 mobile app privacy policies. The technique yields 41% of all subsumption and equivalence axioms in a manually constructed GT ontology with an average precision=0.95 and recall=0.51.

The automated technique requires analysts to code each lexicon phrase using a 1-level, 5-type typology. This step is significantly less burdensome than performing *n* pairwise comparisons for *n*-phrases to manually identify these axioms. For example, a 355-phrase lexicon has a total 62,853 pairwise comparisons, and the automated technique reduces this space by at least 7,719 comparison, and by a total 21,163 comparisons when including the 510 tacit classes generated by the technique to fill gaps in the lexicon.

The typology types can be applied independently to each phrase word, which provides minimal semantic information to distinguish when to vary phrases to infer hypernyms, meronyms and synonyms. The POS tags may provide a means to automate this typing; however, the tags alone cannot determine when a word in a phrase is the phrase head. For example, in "Android id," the word "Android" cannot be recognized as a modifier of “id” from the “NN NN” tag sequence; the “NN” is a common POS tag for technology, as well as modifiers. However, the word “identifying” in the phrase “identifying information” tagged “VBG NN,” may be perceived as an event due to the POS tag, which is a verb. Other VBG-tagged words that are typed as events include advertising, forwarding, and referring.

When comparing the prospective and GT ontologies, we recognized some axioms in the GT ontology that are not consistent with the heuristics and therefore are identified as FNs in the analysis. The GT ontology is manually constructed and is highly influenced by the ontologist. We believe that the automation of ontology construction improves consistency in the final ontology by eliminating some human error due to fatigue and recency effects.

In future work, we envision a number of extensions to improve the method and technique. Within the automated method output, we further discovered 171 meronyms that we classified into six different meronym relationships proposed by Gretsl and Prinnebow (Gerstl & Pribbenow, 1996) (Pribbenow, 1997). In future work, we plan to refine the GT ontology to distinguish among these meronym relationships, and to refine the technique to discover these relationships semi-automatically. We also envision expanding the knowledge base to include relationships among concepts, e.g., “phone” is a kind of “device,” which would enable new rules to infer additional axioms over a larger number of concept name variants. Moreover, we plan to conduct a saturation study with a larger privacy policy lexicon acquired from new privacy policies across multiple domains.

Acknowledgement

We thank Jaspreet Bhatia, Rocky Slavin, and Xiaoyin Wang for their annotations of the 50 mobile app policies, and the CMU RE Lab for their helpful feedback. This research was supported by NSF CAREER #1453139, NSA #141333, NSF #1330596, and NSF #0964710.

References

Anton, A. I., & Earp, J. B. 2004. *A requirements taxonomy for reducing web site privacy vulnerabilities.* Requirements Engineering, 9(3), 169-185.

Bhatia, J., Evans, M., Wadkar, S., Breaux, T.D. 2016. Automated extraction of regulated information types using hyponymy relations. IEEE W'shp AI & Req'ts Engr.

Bhatia, J., & Breaux, T. D. 2015. *Towards an information type lexicon for privacy policies.* Requirements Engineering and Law (RELAW), 2015 IEEE Eighth International Workshop on, (pp. 19-24).

Bradshaw, J., Uszok, A., Jeffers, R., Suri, N., Hayes, P., Burstein, M., . . . others. 2003. *Representation and reasoning for DAML-based policy and domain services in KAoS and Nomads.* Proceedings of the second international joint conference on Autonomous agents and multiagent systems, (pp. 835-842).

Breaux, T. D., & Baumer, D. L. 2011. *Legally “reasonable” security requirements: A 10-year FTC retrospective.* Computers \& Security, 30(4), 178-193.

Breaux, T. D., Hibshi, H., & Rao, A. 2014. *Eddy, a formal language for specifying and analyzing data flow specifications for conflicting privacy requirements.* Requirements Engineering, 19(3), 281-307.

Breaux, T. D., Smullen, D., & Hibshi, H. 2015. *Detecting repurposing and over-collection in multi-party privacy requirements specifications.* 2015 IEEE 23rd international requirements engineering conference (RE), (pp. 166-175).

Breaux, T., & Schaub, F. 2014. *Scaling Requirements Extraction to the Crowd: Experiments on Privacy Policies.* 22nd IEEE International Requirements Engineering Conference (RE'14), (pp. 163-172).

Corbin, J., & Strauss, A. 2014. *Basics of qualitative research: Techniques and procedures for developing grounded theory.* Sage publications.

Fleiss, J. L. 1971. *Measuring nominal scale agreement among many raters*. Psychological bulletin, 76(5), 378.

Gerstl, P., & Pribbenow, S. 1996. *A conceptual theory of part-whole relations and its applications.* Data \& Knowledge Engineering, 20(3), 305-322.

Harris, K. D. 2013. *Privacy on the Go: Recommendations for the Mobile Ecosystem.*

Hearst, M. A. 1992. *Automatic acquisition of hyponyms from large text corpora.* Proceedings of the 14th conference on Computational linguistics-Volume 2, (pp. 539-545).

Hecker, M., Dillon, T. S., & Chang, E. 2008. *Privacy ontology support for e-commerce.* IEEE Internet Computing, 12(2), 54-61.

Horrocks, I., Sattler, U., & Tobies, S. 1999. *Practical reasoning for expressive description logics.* International Conference on Logic for Programming Artificial Intelligence and Reasoning, (pp. 161-180).

Huang, C.-r. 2010. *Ontology and the lexicon: a natural language processing perspective.* Cambridge University Press.

Kagal, L., Finin, T., Paolucci, M., Srinivasan, N., Sycara, K., & Denker, G. 2004. *Authorization and privacy for semantic web services.* IEEE Intelligent Systems, 19(4), 50-56.

Martin, J. H., & Jurafsky, D. 2000. *Speech and language processing.* International Edition, 710.

Miller, G. A. 1995. *WordNet: a lexical database for English.* Communications of the ACM, 38(11), 39-41.

Postman, L., & Phillips, L. W. 1965. *Short-term temporal changes in free recall.* Quarterly journal of experimental psychology, 17(2), 132-138.

Potts, C., & Newstetter, W. C. 1997. *Naturalistic inquiry and requirements engineering: reconciling their theoretical foundations.* Requirements Engineering, 1997., Proceedings of the Third IEEE International Symposium on, (pp. 118-127).

Pribbenow, S. 1997. *What's a part? On formalizing part-whole relations.* Foundations of Computer Science, (pp. 399-406).

Saldaña, J. 2015. *The coding manual for qualitative researchers.* Sage.

Slavin, R., Wang, X., Hosseini, M. B., Hester, J., Krishnan, R., Bhatia, J., . . . Niu, J. 2016. *Toward a framework for detecting privacy policy violations in android application code.* Proceedings of the 38th International Conference on Software Engineering, (pp. 25-36).

Smith, A. 2015. *U.S. Smartphone Use in 2015.* PEW Research Center.

Snow, R., Jurafsky, D., & Ng, A. Y. 2004. *Learning syntactic patterns for automatic hypernym discovery.* Advances in Neural Information Processing Systems 17.

Syed, Z., Padia, A., Finin, T., Mathews, L., & Joshi, A. 2016. *UCO: A Unified Cybersecurity Ontology*. Proceedings of the AAAI Workshop on Artificial Intelligence for Cyber Security.

Uschold, M., & Gruninger, M. 1996. *Ontologies: Principles, methods and applications.* The knowledge engineering review, 11(02), 93-136.

Wilson, S., Schaub, F., Dara, A. A., Liu, F., Cherivirala, S., Leon, P. G., . . . others. 2016. *The Creation and Analysis of a Website Privacy Policy Corpus.*