Automated Extraction of Regulated Information Types using Hyponymy Relations

Jaspreet Bhatia[1], Morgan C. Evans[2], Sudarshan Wadkar[1] and Travis D. Breaux[1]
Institute for Software Research, Carnegie Mellon University[1]
Pittsburgh, Pennsylvania, United States
Bard College [2]
Annandale-on-Hudson, New York, United States
jbhatia@cs.cmu.edu, me4582@bard.edu, swadkar@cs.cmu.edu, breaux@cs.cmu.edu

***Abstract*—Requirements analysts can model regulated data practices to identify and reason about risks of non-compliance. If terminology is inconsistent or ambiguous, however, these models and their conclusions will be unreliable. To study this problem, we investigated an approach to automatically construct an information type ontology by identifying information type hyponymy in privacy policies using Tregex patterns. Tregex is a utility to match regular expressions against constituency parse trees, which are hierarchical expressions of natural language clauses, including noun and verb phrases. We discovered the Tregex patterns by applying content analysis to 15 privacy policies from three domains (shopping, telecommunication and social networks) to identify all instances of information type hyponymy. From this dataset, three semantic and four syntactic categories of hyponymy emerged based on category completeness and word-order. Among these, we identified and empirically evaluated 26 Tregex patterns to automate the extraction of hyponyms from privacy policies. The patterns identify information type hypernym-hyponym pairs with an average precision of 0.83 and recall of 0.52 across our dataset of 15 policies.**

***Index Terms*—Hyponym, hypernym, natural language processing, ontology, privacy policy, compliance.**

I. Introduction

In information systems, personal privacy relates to how personal information is collected, used, and shared. Regulatory bodies enforce these concerns in laws, such as the E.U. Directive 95/46/EC and the Health Insurance Portability and Accountability Act (HIPAA) Privacy Rule. To help requirements analysts design legally compliant and privacy-preserving systems, new methods have been proposed. For example, Ghanavati's compliance framework for business process specification as applied to Canada's privacy health law [8], Maxwell and Anton's production rule system for extracting legal requirements from privacy law [17], Breaux et al.'s Eddy language for asserting privacy principles [3, 5], and Paja et al.'s STS-ml tool for analyzing privacy and security requirements in socio-technical systems [19]. These methods and the problems that they address present a challenge to requirements analysts: what does the category of personal information formally consist of, in order to infer the consequences of collecting and sharing such information?

In this paper, we report results from developing and evaluating an automated method for extracting an information ontology from privacy policies. Privacy policies are posted at most websites, and frequently required by best practice or law, e.g., HIPAA and the Gramm-Leach Bliley Act. These policies describe the data practices of online services and frequently include data practice descriptions for physical locations where services are rendered. In general, a privacy policy describes what information is collected, how it is used, and with whom it is shared. These descriptions frequently include examples that illustrate relevant kinds of information, called sub-ordinate terminology or *hyponyms*. When interpreting these policies and laws, statements that regulate an information type could logically regulate any hyponyms, for example, any restrictions on "contact information" could also be applied to "email address."

The contributions of this paper are three-fold: first, we identify a taxonomy of hyponymy patterns that describes the complete set of hyponyms manually identified among 15 privacy policies; second, we formalize these patterns using Tregex, a tree regular expression language for matching constituency parse trees [15]; and third, we report the number of information types covered by these patterns, called coverage, when compared to a lexicon of information types extracted from the same policies.

The remainder of the paper is organized as follows: in Section II, we review background concepts and related work; in Section III, we present our approach to identifying hyponymy automatically; in Section IV, we present results and the evaluation of our approach; and in Section V, we discuss our results and future work.

II. Background and Related Work

We now review hyponymy in natural language, Tregex and related work.

Hyponyms are specific phrases that are sub-ordinate to another, more general phrase, which is called the *hypernym* [11]. Speakers and readers of natural language typically use the linking verb phrase *is a kind of* to express the relationship between a hyponym and hypernym, e.g., a GPS location is a kind of real-time location. Other semantic relationships of interest include *meronyms*, which describe a part-whole relationship, *homonyms*, which describe the same word that has two unrelated meanings, and *polysemes*, which describe the same word with two related meanings [11]. A popular online lexical database that contains hyponyms is called WordNet [18].

Marti Hearst first proposed a set of six lexico-syntactic patterns to identify hyponyms in natural language texts using noun phrases and regular expressions [9]. The patterns are domain independent and include the indicative keywords "such as," "including," and "especially," among others. The Hearst approach applies grammar rules to a unification-based constituent analyzer over part-of-speech tags to find noun phrases that match the pattern, which are then checked against an early version of WordNet for verification [9]. The approach was unable to work for meronymy in text.

Snow et al. applied WordNet and machine learning to a newswire corpus to identify lexico-syntactic patterns and hyponyms [23]. Their approach includes the six Hearst patterns and resulted in a 54% improvement over WordNet. Unlike Hearst and Snow et al., information types are rarely found in WordNet: among the 1300 information types used in our approach described herein, only 17% of these phrases appear in WordNet, and only 19% of the phrases matched by our hyponymy patterns appear in WordNet. This means that requirements analysts who want to find the category of an information type, or find the members of an information category, will be unlikely to find these answers in WordNet. Our work aims to identify these hyponyms for reuse by requirements analysts in future projects.

Hyponym and hypernym identification relates to theories of categorization, which is studied in cognitive science. This includes Eleanor Rosch's category theory and Tversky's formal approach to category resemblance. Rosch introduced category theory that aims to explain how abstractions relate to one another in taxonomies [20]. In these taxonomies, the more inclusive a category is, the higher that category appears in the taxonomy. Higher-level categories are hypernyms, which contain lower-level categories or hyponyms. In addition, Rosch characterizes categories by the features they share, and she uses this designation to introduce the concept of *cue validity*, which is the probability that a cue *x* is the predictor of a category *y*. Categories with high cue validity are what Rosch calls *basic-level categories*. In our analysis, some hyponyms may also be basic-level categories, however, the features that define such categories are not typically explicit in policy text, and may be tacit knowledge.

An important assumption in Rosch's definition of taxonomy is that each category can at most be a member of one other category. Information type names violate this assumption, because an "e-mail address" can be classified as both "login information" and "contact information," depending on how the e-mail address is used in an information system. Thus, information types may be more amenable to mathematical comparison using Tversky's *category resemblance*, which is a measure in which disjoint categories combine when their shared features outweigh their unshared features [26]. Category resemblance also accounts for asymmetry in similarity [26], which may account for differences arising from confusion among hyponymy, meronymy, homonymy and polysemy. Our approach to extract hyponyms from text does not account for these measured interpretations by Rosch and Tversky, but instead relies on policy authors' authority to control meaning.

In our approach, we use Tregex, which is a utility developed by Levy and Andrew to match constituency parse trees [15]. Constituency parse trees are constructed from part-of-speech (POS) tagged sentences in which each word is tagged with a POS tag, such as a noun, verb, adjective, or preposition tag, among others. Tregex has been used to generate questions from declarative sentences [10], to evaluate text summarization [25], to characterize temporal requirements [16], and to generate interpretations of regulatory policies [13].

While natural language processing (NLP) of requirements texts can scale analysis to large corpora, the role of NLP should not be overstated, since it does not account for human interpretation [21]. Jackson and Zave argue that requirements engineering is principally concerned with writing accurate software specifications, which require explicit statements about domain phenomena [12]. While significant work has been done to improve specification, a continuing weakness is that problems are frequently formalized using low-level programming concepts (classes, data, and operations) as opposed to using richer, problem-oriented ontologies [14]. In this paper, we investigate an approach for extracting an information type ontology from higher-order descriptions of information systems embodied in privacy policies. We believe these ontologies can improve how we reason about and analyze privacy requirements for web-based and mobile information systems.

III. Automated Hyponymy Extraction

We now introduce our research questions, followed by our research method based on content analysis and Tregex.

RQ1. What are the different ways to express hyponymy in privacy policies, and what categories emerge to characterize the linguistic mechanisms for expressing hyponymy?

RQ2. To what extent can constituency parsers be used to extract hyponymy from privacy policies?

RQ3. What percentage of information types available in privacy policies can be extracted by applying the hyponymy patterns to these policies?

Figure 1 presents an overview of our approach to answer the research questions. During steps 1 and 2, the analyst prepares the input text to the NLP tools used in steps 3 and 4, and to the crowdworker platform in step 6, which is based on Amazon Mechanical Turk (AMT).

Steps 1-2 are performed manually by an analyst, once for each policy, which requires only a few minutes per policy. In step 1, the input text begins as a text file, which can be extracted from HTML or a PDF document. In step 2, the analyst itemizes the text into paragraphs consisting of 50-120 words, while ensuring that each paragraph's context remains undivided. This invariant can lead to paragraphs that exceed 120 words, which is balanced by smaller 50-60 word paragraphs. The 120-word limit is based on the average time required by one crowdworker to identify information types in step 6, which averages 60 seconds.

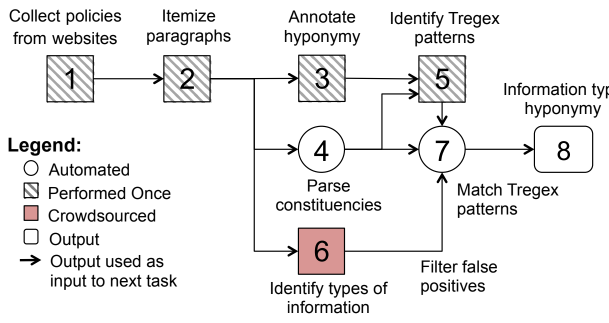


Fig. 1. Hyponymy Extraction Framework

In step 3, two or more analysts perform content analysis on the same 120-word paragraphs from step 2 to manually identify and categorize hyponymy in privacy policy text. The analysts meet periodically to agree on heuristics and guidelines for annotating hyponymy, before combining the hyponymy annotations with corresponding constituency parses from step 4 to infer matching Tregex patterns in step 5. The Tregex patterns are identified in a manual, interpretive process performed once: the patterns are then used in an automated step 7 to find hyponymy relationships.

The Tregex patterns are generic and do produce false positives. To filter out false positives, we use a crowdworker lexicon constructed from crowdworker annotations produced in step 6. If a Tregex pattern matches a phrase that has not been annotated as an information type, that match is discarded as a false positive. We describe each of these steps in detail in the following sub-sections.

A. Annotating Hyponymy

Research question RQ1 asks how hyponymy appears in privacy policies "in the wild." To answer RQ1, we selected 15 privacy policies across three domains: shopping, telecom, and social networking (see Table I). These policies are part of a convenience sample, although, we include a mix of shopping companies who maintain both online and "brick-and-mortar" stores, and we chose websites with the largest number of users in 2015. Table I presents the 15 policies by category and date last updated.

The policies are first prepared by removing section headers and boilerplate language that does not describe relevant data practices, before saving the prepared data to an input file for an Amazon Mechanical Turk (AMT) task, as described by steps 1 and 2 in Figure 1. The task employs an annotation tool developed by Breaux and Schaub [4], which allows two or more annotators to select relevant phrases matching a category. The annotators are asked to annotate three types of phrases for each hyponymy relationship identified: a *hypernym phrase*, which describes the general category phrase; one or more *hyponym phrases*, which describe members of the category; and any *keyword*, which signals the hyponymy relationship. For example, in Figure 2, the annotated phrase "personal information" is the hypernym, which is followed by the keywords "for example," which indicate the start of a clause that contains the hyponyms "name," "address" and "phone number."

TABLE I. PRIVACY POLICY DATASET FOR HYPONYMY STUDY

Company's Privacy Policy	Industry Category	Last Updated
Barnes and Noble	Shopping	05/07/2013
Costco	Shopping	12/31/2013
Lowes	Shopping	04/25/2015
Overstock	Shopping	01/09/2013
Walmart	Shopping	9/17/2013
AT&T	Telecom	09/16/2013
Charter Comm.	Telecom	05/04/2009
Comcast	Telecom	03/01/2011
Time Warner	Telecom	09/2012
Verizon	Telecom	10/2014
Facebook	Social Networking	4/9/2013
Kik	Social Networking	9/22/2014
LinkedIn	Social Networking	10/23/2014
Snapchat	Social Networking	11/17/2014
WhatsApp	Social Networking	7/7/2012

The annotation process employs two-cycle coding [22]. In the first cycle, the policies are annotated to identify the prospective hyponym patterns, after which the second cycle is applied to group these patterns into emergent categories. The two-cycle coding process begins with an initial set of five policies, during which guidelines and examples are developed by the annotators to improve consistency and to support replication. Next, the annotators meet to discuss their initial results, to reconcile differences and to refine the guidelines. After agreeing on the guidelines and initial categories, the annotators annotate the remaining policies, before meeting again to reconcile disagreements.

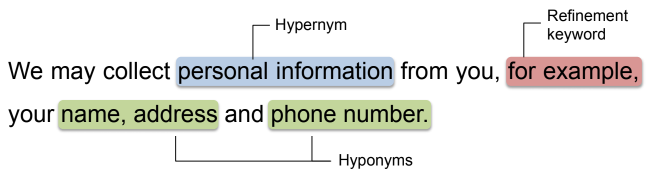
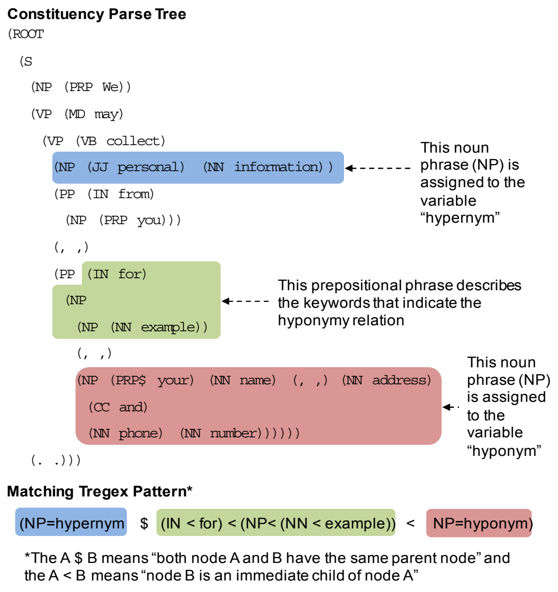

Hypernym
Refinement keyword
We may collect personal information from you, for example,
your name, address and phone number.
Hyponyms

Fig. 2. Hyponymy Annotations

B. Identifying Tregex Patterns

Tregex is a language for matching subtrees in a constituency parse tree [15]. The Tregex patterns are created by examining the parse tree for each annotated sentence in a hyponymy category. Figure 3 presents the constituency parse tree for the sentence from Figure 2. The colors in the figure show which part of the parse tree matches which part of the Tregex pattern. In the parse tree, the root tree node labeled ROOT appears in the upper left-hand corner with a single, immediate child labeled S; each child is indented slightly to the right under the parent, and siblings are indented equidistant from the left-hand side of the figure. The matching Tregex pattern below the parse tree has three parts: a noun phrase (NP) that is assigned to a variable named "hypernym" via the equal sign (in blue), followed by a dollar sign that indicates a sibling pattern, which is the keyword phrase (in green), followed by a less-than sign that indicates an immediate child node, which is another NP assigned to the variable "hyponym" (in red). The ability to reference words or phrases by their position in the tree, such as immediate child node, or their breadth with respect to other words or phrases, such as sister node, allows us to disregard modifying phrases that may otherwise have been extracted using flat annotations solely based on POS-tags. In Figure 3, for example, the noun phrase (NP) "personal information" is separated by the constituency parser from the NP in the sister, prepositional phrase (PP) that we wish to exclude. The Tregex patterns are thus simpler to express than regular expressions over POS-tags alone. Tregex provides a means to answer RQ2 by expressing patterns that match the annotated hypernyms and hyponyms and their lexical coordination by the keywords.


Constituency Parse Tree
This noun phrase (NP) is assigned to the variable "hypernym"
This prepositional phrase describes the keywords that indicate the hyponymy relation
This noun phrase (NP) is assigned to the variable "hyponym"
Matching Tregex Pattern*
*The A $ B means "both node A and B have the same parent node" and the A < B means "node B is an immediate child of node A"

```
(ROOT
 (S
  (NP (PRP We))
  (VP (MD may)
   (VP (VB collect)
    (NP (JJ personal) (NN information))
    (PP (IN from)
     (NP (PRP you)))
    (, ,)
    (PP (IN for)
     (NP
      (NP (NN example))
      (, ,)
      (NP (PRP$ your) (NN name) (, ,) (NN address)
       (CC and)
       (NN phone) (NN number))))))
  (. .)))
```

```
(NP=hypernym $ (IN < for) < (NP< (NN < example)) < NP=hyponym)
```

Fig. 3. Tregex Pattern Matcher

The development of Tregex patterns is a balance between generally characterizing the lexical relationships among words in a pattern, and specializing the pattern to avoid false positives. The pattern shown in Figure 3, which matches information type hyponyms after the "for example" keywords, can also match a data purpose hyponym: e.g., "we share your personal information for marketing, for example, product and service notifications." To automatically filter out such hyponyms that are not information type hyponyms, we use a lexicon constructed from crowd sourced information type tasks, described by Breaux and Schaub [4]. In the information type tasks, the crowdworkers are asked to annotate all information types in a given paragraph. For instance, in the privacy statement "*We collect your personal information such as your name, and address…",* the crowdworkers would have annotated the phrases "personal information", "name" and "address" as information types.

We evaluated our Tregex patterns in two ways: first we evaluated each pattern to match the hyponymy annotations per statement; and second we evaluated the collection of all patterns by the number of unique hypernym-hyponym pairs that were identified across the whole collection of 15 policies. The first evaluation concerns the ability of the Tregex patterns to reproduce the human annotators' annotations, whereas the second evaluation measures the overall goal to extract an ontology across all the policies, recognizing the various ways that a hypernym-hyponym pair can be expressed.

C. Ontological Completeness

The question RQ3 asks what percentage of all the information types found in the privacy policies can be extracted by applying the Tregex patterns to privacy statements. To answer RQ3, we developed three types of lexicons – crowdworker lexicon, annotator lexicon and Tregex lexicon. The *crowdworker lexicon* consists of all the information types in our dataset of 15 privacy policies (see Table I). For the construction of this lexicon, we use the entity extractor developed by Bhatia and Breaux [1], which takes as input the crowd sourced tasks for each policy, where the crowdworkers have annotated all the information types in the policies [4]. The *annotator lexicon* is constructed by using the entity extractor on the analysts' hyponymy annotations described in Section III.A. The *Tregex lexicon* is constructed using the entity extractor on the information type hyponymy identified by the Tregex patterns, as described in Section III.B. We compare the *crowdworker lexicon, the annotator lexicon* and the *Tregex lexicon* to understand what percentage of information types can be organized ontologically using hyponymy patterns.

IV. Evaluation and Results

We now describe our results from the content analysis, Tregex pattern development and the lexicon comparisons.

A. Hyponymy Taxonomy from Content Analysis

The first and second authors annotated the 15 policies shown in Table I. This process consumed 235 and 282 hours for each annotator, respectively, and yielded 232 annotated instances of hyponymy after final reconciliation. The guidelines that were developed can be summarized as follows: only annotate information type noun phrases; annotations should not span more than a single sentence and should include any modifying prepositional or verb phrases that qualify the information type; annotate the pronoun as a hypernym if the actual information type is not present in the sentence (e.g. this); and annotate all the noun phrases together, if the noun phrase is an enumeration.

After the second cycle coding, the two annotators categorized the hyponymy relationships first based on semantics as suggested by the refinement keywords, and the second based on the relative order of the hypernym (H), keyword (K) and hyponym (O) in the privacy policy text. The semantic categories are based on the keywords used to indicate hyponymy. The resulting categories are as follows:

- *Incomplete Refinement (Inc.)*: The keywords indicate the hyponymy is an incomplete phrase subset for the given hypernym. The keywords "such as" and "including" indicate that subsequent hyponyms are exemplary.

- *Complete Refinement (Com.)*: The keywords indicate that the hyponyms are the complete list that belongs to the hypernym. The keywords, "consists of" and "i.e." indicate that subsequent hyponyms are complete for the hypernym.
- *Implied Refinement (Imp.)*: The refinement keyword is a punctuation such as a colon (:) or dash (-) and indicates that there is an implied hyponymy.

The resulting syntactic categories are defined as follows:

- *HKO* – The hypernym precedes the keywords, followed by the hyponym. This pattern is predominantly used to illustrate examples (hyponyms) of leading technical words (the hypernym).
- *OKH* – The hyponym precedes the keyword, followed by the hypernym. This category describes lists, in which the last term generalizes the preceding terms.
- *HO* – The hypernym precedes the hyponym with no keyword. This category appears when hypernymy is the section header, followed *implied* hyponyms in the section; no keywords explicitly indicate the hyponymy.
- *KHO* – The keyword precedes the hypernym, followed by the hyponym. This category is rare, and uses a colon to separate the hypernym from a list of hyponyms.

We measured the degree of agreement above chance using Fleiss' Kappa [7] for the hyponym categories from the second-cycle coding. Each hyponymy instance is assigned a semantic category and a syntactic category. The Kappa was computed on the category composition. For example, a hyponymy that belongs to the incomplete semantic category and HKO syntactic category is assigned to the category combination of {Inc.-HKO}. The Kappa for mappings from annotations to hyponym categories and the two annotators was 0.96 which is a high degree of agreement above chance.

Table II and III presents the keyword taxonomies for the semantic and syntactic categories, respectively, including the *Category*, the *Refinement Keywords* that help detect the hyponymy, and the proportion of annotations in the category across all 15 policies (*Freq.*). The most frequent category among the semantic categories was *incomplete refinement*.

TABLE II. KEYWORD TAXONOMY FOR SEMANTIC CATEGORIES

Category	Refinement Keywords	Freq.
Incomplete Refinement	such as, such, include, including, includes, for example, e.g., like, contain, (and|or|any | as well as any| certain) other, concerning, relating to,, is known as, classifies as	94.40%
Complete Refinement	consists of, is, i.e., either, constitute, of your, following types of	3.45%
Implied Refinement	(), :, -, . (section header)	2.16%

TABLE III. KEYWORD TAXONOMY FOR SYNTACTIC CATEGORIES

Category	Refinement Keywords	Freq.
HKO	such as, such, including, for example, include, includes, concerning, is, e.g., like, i.e., of your, contain, relating to, that relates to, generally not including, consists of, concerning, either, ( ), :, -	84.05%
OKH	(and|or|any| as well as any| certain) other, constitute, as, and any other, is known as, classifies as	14.66%
HO	None	0.86%
KHO	following types of	0.43%

H: Hypernym, O: Hyponym, K: Keyword

Table IV presents the frequency of annotations per category combination across the 15 policies. The most frequent category combination was Inc.-HKO, with 80.6% of all annotations belonging to this category. Saturation was reached after annotating six policies, after which we found no additional refinement keywords or categories.

TABLE IV. FREQUENCY OF HYPONYMY CATEGORIES

Syntactic Categories	Semantic Categories			
	Inc.	Com.	Imp.	Total
HKO	187	5	3	195
OKH	32	2	0	34
HO	0	0	2	2
KHO	0	1	0	1
Total	219	8	5	232

H: Hypernym, O: Hyponym, K: Keyword; Inc.: Incomplete Refinement, Com.: Complete Refinement, Imp.: Implied Refinement

B. Tregex Pattern Evaluation

We identified a total of 26 Tregex patterns, which can be used to automatically identify hyponymy in privacy policies. Due to space limitations, we only present an example subset of Tregex patterns in Table V. The HO syntactic category, which has no keywords, cannot be reliably characterized by a high precision pattern, i.e., low false positives.

TABLE V. TREGEX PATTERNS

Hyponym Category	Tregex Pattern Example	# Tregex Patterns
HKO	NP=hypernym $ (VP < (VBZ < includes) < NP=hyponym)	22
OKH	NP < (NP=hyponym $. ((CC < (or\|and)) ?$. other) $. NP=hypernym)	3
KHO	NP < (NP < ((NP < (JJ < following) < (NNS < types)) $. (PP < (IN < of) < NP=hypernym)) $.. NP=hyponym	1

We evaluate our Tregex pattern approach in two ways, as described in Section III.B. Table VI presents the first evaluation precision and recall in terms of the automated hyponym extraction results compared to the annotator annotations for the 15 policies. The identified instance of hyponymy is counted as a true positive (TP), only if the hypernym and the hyponym both match the annotator annotations. Otherwise, it is counted as a false positive (FP). The results in Table VI were computed by using the crowdworkers' information type annotations as a means to filter out FPs as described in Section III.B; without this filtering, the average precision drops from 0.73 to 0.54.

TABLE VI. EVALUATIONS OF TREGEX PATTERNS

Privacy Policy	Precision	Recall
Barnes and Noble	0.69	0.47
Costco	0.86	0.67
Lowes	0.86	0.75
Overstock	0.67	0.40
Walmart	0.86	0.86
AT&T	0.64	0.56
Charter Comm.	0.80	0.63
Comcast	0.50	0.40
Verizon	0.69	0.56
Time Warner	0.43	0.50
Facebook	0.73	0.63
Kik	0.88	0.54
LinkedIn	0.90	0.60
Snapchat	0.75	0.50
WhatsApp	0.75	0.60
Average	**0.73**	**0.58**

For the second evaluation, we compiled an ontology of hypernym-hyponym pairs identified by the Tregex patterns and compared these to the pairs found by the annotators across all 15 policies. The Tregex patterns yielded 209 hypernym-hyponym pairs compared to the 333 pairs identified by the annotators; there are 173 pairs in common. The average precision is 0.83, and average recall is 0.52.

We analyzed FPs and false negatives (FNs) produced by the Tregex patterns to explain the low recall. The Tregex uses the Stanford Parser to tag part-of-speech (POS), before building the parse trees. Incorrect POS-tags are one source of misidentified hypernyms. The parse tree may also be inaccurate due to syntactic ambiguity, in which a modifier phrase is incorrectly attached to a preceding noun phrase. For example, the statement, "So for those we develop a more precise estimate of location by associating the serving cell tower ID with other information, like the latitude and longitude of the tower…", the Stanford Parser attaches the noun phrase, "precise estimate of location" to the modifier phrase "like the latitude and longitude." Our Tregex pattern approach identifies this attachment to be the hypernym of the modifier, as opposed to "other information." The annotators are able to disambiguate such attachments based on context and their domain knowledge. However, incorrect parse trees generated by the Stanford Parser yield unique and ungeneralizable patterns, which prevents pattern saturation.

Another reason for FNs is nested hyponymy relations, wherein a combination of two or more patterns is required to accurately extract information types. Statements with nested hyponymy comprised less than ten of the total number of hyponymy relations, and thus we chose to exclude patterns for nested hyponymy from our results to yield lower recall.

Our true positives also include incomplete identification of the hypernyms due to the presence of anaphora pronouns. For example, the sentences "We collect your personal information. *This* includes your name, address…" contains the pronoun "this," which refers to the noun phrase, "personal information" in the previous sentence. Our automated approach is limited, and does not contain a co-reference resolution engine that is able to identify such instances of pronouns.

C. Comparison to Lexicon

As described in Section III.C, we constructed our *crowdworker lexicon* using the entity extractor [1] on the crowdsourced tasks [3] for the 15 policies in our dataset (see Table I). This dataset had a total of 1300 unique phrases. The *annotator lexicon*, which was constructed using the entity extractor on the annotator hyponymy annotations, contains 461 phrases. The *crowdworker lexicon* contains 429 out of these 461 phrases, which converts to 33% of the total information types identified by the crowdworkers in the privacy policies using the annotator annotations. The difference in 32 phrases were false negatives (FN) that were identified by the annotators during hyponymy annotations, but were missed by the crowdworkers during the information type annotation tasks. The FN information types contained some information types that had uncommon meaning, for example "public profile" and were difficult to identify. The FNs also had information types that were different forms of the information types already existing in the crowdworker lexicon, for instance, "new personal information" and "similar account information."

The *Tregex lexicon*, constructed by applying the entity extractor to the hyponymy identified by the Tregex patterns yielded 325 phrases. The *Tregex lexicon* shared 293 phrases with the *crowdworker lexicon*, which yields 23% of the information types identified by crowdworkers in the privacy policies. The Tregex patterns yielded 32 phrases that are false positives (FP), meaning that they were identified by the Tregex, but not by the crowdworkers. On further analysis of the FP phrases, we found that six of these phrases could be included as true positives (TP), and were instead missed by the crowdworkers. The phrase, "aggregate demographic information," for example, is present in the *Tregex lexicon*, but is missing from the *crowdworker lexicon*.

We found 285 phrases that were shared between the *Tregex lexicon* with the *annotator lexicon*.

V. DISCUSSION AND FUTURE WORK

We now discuss our results and their impact on improving awareness of privacy practices and for privacy goal extraction. Hyponymy is one among other relationships, such as meronymy and synonymy, that is used to describe the relationships between various phrases in an ontology. Using our approach, we were able to identify hypernym-hyponym pairs with an average precision of 0.83 and average recall of 0.52, as compared to the pairs identified by the annotators. We believe that this is the first step towards automating the construction of an information type ontology for privacy policies, by reducing the search space of possible relationships that exist between two phrases in a given lexicon. For the 1300 information types identified by crowdworkers in the 15 policies, there are 1.69M such possible comparisons needed to construct a complete ontology. In addition, to determine the relationship between the 293 shared terms between the Tregex lexicon and the crowdworker lexicon, we would have needed 42,778 paired comparisons, which we identified automatically

During our analysis, we observed that the vocabulary used in the privacy policies is very different from the vocabulary used in the popular lexical database such as WordNet, as only 17% of the information types we found in our dataset of 15 policies were present in the WordNet.

One limitation of our approach is that the extracted hypernyms at most describe the information types explicitly stated in the policy, and hence do not account for implied information types. For example, the phrase, "personal information includes browsing history" indicates browsing history is personal. This history could include visits to popular news and shopping websites, which is common, it could include sites from which the user's risk for metastatic melanoma could be inferred. Such inferred information types, e.g., potential diseases, unless stated in the policy as a hyponym, will be missing from the results of our approach.

As future work, we envision using the dependency parser to explore the relationships between individual words rather than the arrangement of grouped words from the constituency parser to identify hyponymy relationships among phrases. Furthermore, we envision using our empirically validated hyponymy annotations as a training set for machine learning algorithms to develop domain specific models of hyponymy.

ACKNOWLEDGMENT

We thank the CMU RE Lab, including João Caramujo, Liora Friedberg, Mitra Bokaei Hosseini, Daniel Smullen, and David Widder for their helpful feedback. This research was funded by NSF Frontier Award #1330596, NSF CAREER Award #1453139 and NSA Award #141333.

REFERENCES

[1] J. Bhatia, T.D. Breaux, "Towards an information type lexicon for privacy policies." *IEEE 8th Int'l W'shp Req'ts Engr. & Law*, pp. 19-24, 2015.

[2] J. Bhatia, T.D. Breaux, F. Schaub. "Privacy goal mining through hybridized task re-composition", *ACM Trans. Soft. Engr. Method.*, 25(3): Article 22, 2016..

[3] T.D. Breaux, H. Hibshi, A. Rao. "Eddy, a formal language for specifying and analyzing data flow specifications for conflicting privacy requirements." *Req'ts Engr. J.*, 19(3): 281-307, 2014.

[4] T.D. Breaux, F.Schaub, "Scaling requirements extraction to the crowd: experiments on privacy policies." *IEEE 22nd Int'l Req'ts Engr. Conf.*, pp. 163-172, 2014.

[5] T.D. Breaux, D. Smullen, H. Hibshi. "Detecting repurposing and over-collection in multi-party privacy requirements specifications." *IEEE 23rd Int'l Req'ts Engr. Conf.,* pp. 166-175, 2015.

[6] J. Corbin, A.S. Strauss, (2007) *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, 3rd ed. Sage Publications.

[7] J.L. Fleiss, "Measuring nominal scale agreement among many raters," *Psychological Bulletin*, n. 76, pp. 378-382.

[8] S. Ghanavati, *Legal-URN framework for legal compliance of business processes.* Ph.D. Thesis, University of Ottawa, 2013.

[9] M. A. Hearst, "Automatic acquisition of hyponyms from large text corpora," in proceedings of the 14th conference on Computational linguistics - Volume 2 (COLING '92), Vol. 2. Association for Computational Linguistics, 1992, pp. 539-545. DOI=http://dx.doi.org/10.3115/992133.992154

[10] M. Hellman, N.A. Smith, "Good question! statistical ranking for question generation," *Annual Conf. North Amer. Chapter of the. ACL*, pp. 609–617, 2010.

[11] D. Jurafsky, J.H. Martin, *Speech and language processing.* Prentice Hall, 2009.

[12] M. Jackson, P. Zave, "Domain descriptions," *IEEE Symp. Req'ts Engr*, pp. 56-64, 1993.

[13] G. Koliadis, N.V. Desai, N.C. Narendra, A.K. Ghose. "Analyst-mediated contextualization of regulatory policies," *IEEE Int'l Conf. Services Comp.,* pp. 281-288, 2010.

[14] A. van Lamsweerde, "Formal specification: a roadmap," Int'l Soft. Engr. Conf., pp. 147-159, 2000.

[15] R. Levy, G. Andrew. "Tregex and Tsurgeon: tools for querying and manipulating tree data structures." *5th Int'l Conf. on Lang. Res. & Eval.* (LREC 2006).

[16] W. Li, J. Huffman Hayes, M. Truszczynski, "Torwards more efficient requirements formalization," REFSQ, *LNCS* v. 9013, pp. 181-197, 2015.

[17] J. Maxwell, A. Antón. "Developing production rule models to aid in acquiring requirements from legal texts," *17th IEEE Int'l Req'ts Engr. Conf.*, pp. 101-110, 2009.

[18] G. A. Miller. "WordNet: A Lexical Database for English." *Comm. ACM* 38(11): 39-41, 1995.

[19] E. Paja, F. Dalpiaz, P. Giorgini. "Modelling and reasoning about security requirements in socio-technical systems," *Data & Know. Engr.* v. 98, pp. 123-143, 2015.

[20] E. Rosch, "Principles of categorization," Rosch & Lloyd (eds), *Cognition and Categorization*, pp. 27-48, 1978.

[21] K. Ryan, "The role of natural language in requirements engineering," *IEEE Symp. Req'ts Engr*, pp. 240-242, 1993.

[22] J. Saldaña. *The Coding Manual for Qualitative Researchers*, SAGE Publications, 2012.

[23] R. Snow, D. Jurafsky, and A. Y. Ng, "Learning syntactic patterns for automatic hypernym discovery," in Advances in Neural Information Processing Systems, 2005, pp-??

[24] R. Slavin, X. Wang, M.B. Hosseini, W. Hester, R. Krishnan, J. Bhatia, T.D. Breaux, J. Niu. "Toward a Framework for Detecting Privacy Policy Violation in Android Application Code," *ACM/IEEE 38th International Conference on Software Engineering*, pp. 25-36, 2016.

[25] S. Tratz, E.H. Hovy. "Summarization Evaluation Using Transformed Basic Elements." *Text Analytics Conference (TAC-08),* NIST, 2008.

[26] A. Tversky. "Features of similarity." *Psych. Review*, 84(4): 327-352, 1977.