Mining Privacy Goals from Privacy Policies using Hybridized Task Re-composition

JASPREET BHATIA, Carnegie Mellon University
TRAVIS D. BREAUX, Carnegie Mellon University
FLORIAN SCHAUB, Carnegie Mellon University

Privacy policies describe high-level goals for corporate data practices, and regulators require industries to make available conspicuous, accurate privacy policies to their customers. Consequently, software requirements must conform to those privacy policies. To help stakeholders extract privacy goals from policies, we introduce a semi-automated framework that combines crowdworker annotations, natural language typed dependency parses and a reusable lexicon to improve goal extraction coverage, precision and recall. The framework evaluation consists of a five policy corpus governing web and mobile information systems yielding an average precision of 0.73 and recall of 0.83. The results show that no single framework element alone is sufficient to extract goals, however the overall framework compensates for elemental limitations: human annotators are highly adaptive at discovering annotations in new texts, but those annotations can be inconsistent and incomplete; dependency parsers lack sophisticated, tacit knowledge, but they can perform exhaustive text search for prospective requirements indicators; and while the lexicon may never completely saturate, the lexicon terms can be reliably used to improve recall. Lexical reuse reduces false negatives by 41%, increasing the average recall to 0.85. Lastly, crowd workers were able to identify and remove false positives by around 80%, which improves average precision to 0.93.

Categories and Subject Descriptors: **D.2.1 [Software Engineering]:** Requirements/Specifications

General Terms: Design, Languages

Additional Key Words and Phrases: Requirements Extraction, Crowdsourcing, Natural Language Processing, and Privacy.

ACM Reference Format:

Jaspreet Bhatia, Travis Breaux, and Florian Schaub, 2016. Mining Privacy Goals from Privacy Policies using Hybridized Task Re-composition. *ACM Transaction on Software Engineering and Methodology (TOSEM)*. X, X, Article XX (XXX), X pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

Software engineers who design data-intensive systems must increasingly consider the impact of privacy on their software design. Regardless of whether the software is developed using a plan-driven or agile method, software that operates on personal information is often required to be accompanied by a privacy policy, which is a legal document and software artifact that describes consumer data practices. Privacy policies answer important questions about the software's operation, including what personal identifiable information is collected, for what purpose is it used, and with whom is it shared. These policies also serve a U.S. and E.U. regulatory role to

This work is supported by the National Science Foundation, under grant #1330596.

Author's addresses: J. Bhatia, T. D. Breaux, F. Schaub, 5000 Forbes Avenue, Institute for Software Research, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, 15213.

DOI: http://dx.doi.org/10.1145/0000000.0000000

increase data transparency [Reidenberg et al. 2015], and they are used to conduct market surveillance[1] on industry data practices. The transparency benefits of privacy policies include that consumers can learn how companies use their personal data, while software developers can learn how their third party service providers protect or put at risk their users' data. User data may be put at risk, for example, when a third party advertising library acquires that data for use under a more permissive third-party privacy policy.

Methods exist that can be used to help regulators and developers analyze privacy policies. These techniques are based on privacy goals, which describe actions of a system or stakeholder and that are performed on an information type, such as "collecting your e-mail address" or "sharing your location." Goals are also linked to software requirements through goal refinement [Dardenne et al. 1993]. Antón and Earp introduced a privacy requirements taxonomy based on the manual extraction of privacy goals from privacy policies [Antón and Earp 2004]. Their taxonomy enumerates privacy vulnerabilities that cover threats due to over-collection, storage and sharing, among others, which engineers can use to perform privacy risk analyses early in the software design lifecycle. The taxonomy was implemented in the Privacy Goal Management Tool (PGMT) to later discover how ChoicePoint, a data broker, introduced vulnerabilities by aggregating consumer information across multiple services [Otto et al. 2007]. Breaux et al. formalized privacy goals in Description Logic [Breaux et al. 2009], which were adopted by the Eddy privacy specification language for detecting conflicting privacy goals [Breaux et al. 2014] and for tracing data flows across policies in a service composition [Breaux et al. 2015]. Privacy goal tracing across policies is especially important when developers want to know how their third parties may use their users' personal information. Finally, privacy goals appear in other requirements analysis techniques, such as the socio-technical system requirements framework for managing confidentiality goals [Paja et al. 2015] and the Nòmos 3 framework [Ingolfo et al. 2014].

These above techniques show promise in improving privacy policy analysis, yet they all depend on the ability to extract privacy goals from policies, which requires significant time and effort. Based on the number of websites browsed by an average user, and the time required to read a privacy policy, McDonald and Cranor estimate the average user must spend 181–304 hours per year reading privacy policies [McDonald and Cranor 2008]. While developers may need to read fewer privacy policies to assess third party software, the number of services they rely on can still range from tens to hundreds, depending the complexity of the service composition [Lin et al. 2012].

To address the challenge of extracting privacy goals from privacy policies, we describe an empirically evaluated framework that facilitates the extraction of privacy goals from privacy policies using a hybrid combination of crowdsourcing and natural language processing (NLP). In our framework, crowd workers provide phrase-level policy interpretations through small 30-60 second tasks, called micro tasks. To keep the tasks small and affordable, we employ typed dependency parsing based on part-

[1] Since 2013, the Global Privacy Enforcement Network (GPEN), a global coalition of data protection agencies, conducts annual privacy policy sweeps, in which they analyze 1–2,000 privacy policies of online services and mobile apps in a concerted effort.

of-speech (POS) tagging to compose worker annotations into privacy goals, and thus partially automate privacy goal extraction.

The remainder of this paper is organized as follows: in Section 2, we review related work; in Section 3, we introduce our hybrid framework; in Section 4, we present our results from validating the framework with a corpus of five privacy policies; in Section 5, we discuss the threats to validity and in Section 6, we report our discussion and future work.

2. RELATED WORK

We review related work on requirements extraction from text and from crowdsourcing annotations.

2.1 Requirements Extraction

Privacy goals are a semi-formal, canonical representation of what data action is performed on which kind of information. The translation from text to formal and semi-formal specifications has long been a challenge. Abbot et al. were among the first people to propose mining program descriptions from text for object-oriented design [Abbot et al. 1983]. Later, Antón introduced the goal-based requirements analysis method (GBRAM) and heuristics to extract goal specifications from text [Antón 1997]. Goals range from high- and low-level actions to be maintained, achieved and avoided by the system [Dardenne et al. 1993]. Antón and Earp applied GBRAM to mine privacy goals from privacy policies [Antón and Earp 2004], and Breaux and Rao showed how to extract data flow requirements from privacy goals described in privacy policies [Breaux and Rao 2013]. In this prior work, however, the task of extracting goals from policies required training of expert analysts, and high motivation and vigilance that limits the ability to analyze systems of increasing size and complexity. In this paper, we extend this prior work with a narrow focus on privacy goals describing the collection, use, transfer and retention of consumer information with the intent of automating extraction to address the limitations posed by training and expert requirements in prior work.

Majority of the requirements documents and other sources of requirements are written in natural language and consequently numerous efforts have been made to utilize NLP techniques to aid the automated analysis of requirements and to perform other RE tasks [Kof 2004]. These include automatically extracting requirements and goals from text documents, building models from requirements specifications and identifying ambiguity in requirement documents among others. Liu et al. use dependency parsing and pre-defined pattern matching rules to extract relations needed to build i* strategic dependency models [Liu et al. 2014]. We also employ dependency parsing in our approach, however, we seek to maximize precision and recall without curating a rule set. Such rule sets greatly improve precision and recall, but at the cost of continuously needing to update the rule set with each new corpora encountered. Casagrande et al. applied phrase structure grammar to a smart-metering research paper corpus to extract research goals [Casagrande et al. 2014]. Their evaluation of the automated method against 44 manually annotated papers produced a precision = 0.1, and recall = 0.7. In our work, we achieve similar results using dependency parsing—a related NLP technique—with precision near 0.2 and recall near 0.7 (see Table V). However, our results go further to show that the

addition of crowd worker micro data improves precision to 0.64–0.79 and recall to 0.69–0.96 (see Table 7).

A significant challenge in extracting requirements from text is the problem of ambiguity [Kamsties 2006]. Berry et al. introduced the Ambiguity Handbook that describes ambiguity in requirements specifications and legal contracts, and they present several strategies for avoiding and detecting ambiguities [Berry et al. 2003]. Furthermore, object oriented analysis models of the specified system can be used to identify ambiguities and inconsistencies [Popescu et al. 2008]. More recent work has focused on using machine learning algorithms based on heuristics drawn from human judgments, to identify nocuous coordination and anaphoric ambiguities in requirements [Yang et al. 2010; Yang et al. 2011]. This approach still requires human interpretation to detect and resolve ambiguity. In our approach, semantic ambiguity arises when certain verbs indirectly indicate company data practices (e.g., "chatting with friends online" can mean that companies "collect chat logs," the latter being less ambiguous about whether collection occurs). To discover these practices, we rely on crowd workers to disambiguate the text to identify relevant actions and information types. Because crowd workers vary in their ability to perform these inferences, we rely on crowd worker consensus to select likely candidates that we then compare to the expert analysis.

2.2 Crowdsourcing Annotations and Extraction

Crowdsourcing facilitates tackling problems that remain hard to solve with automated methods by leveraging human intelligence, typically provided by non-experts [Quinn and Bederson 2011]. Sabou et al. note that crowdsourcing plays a major role in natural language processing (NLP) as an affordable, large-scale means to acquire corpora and train and evaluate extraction methods [Sabou et al. 2012]. Crowdsourced annotations from non-experts have also been shown to be comparable to expert annotations for certain annotation tasks, such as word similarity, word sense disambiguation and textual entailment recognition [Snow et al. 2008]. Crowdsourcing has also been employed for requirements elicitation: StakeRare uses social networks and collaborative filtering to elicit and prioritize user requirements [Lim and Finkelstein 2012]. Breaux and Schaub studied three tasks to extract privacy goals from policies relying only on untrained crowd workers [Breaux and Schaub 2014]. They show that task-decomposition, which requires splitting the goal extraction task into micro tasks, yields better results with lower cost compared to expert analysts. The framework presented herein is based on this observation and employs NLP-based techniques to recompose the micro task results into privacy goals. We now discuss crowdsourcing in extraction-related tasks and challenges in crowdsourcing text annotations.

In order to leverage the potential of crowdsourcing for annotating and extracting information from natural language text a number of challenges need to be addressed. André et al. [André et al. 2014] identify major challenges in having non-experts (novices) perform the annotations, having a transient workforce, and the need to resolve conflicting (and potentially erroneous) annotations. We now discuss these challenges in detail.

Novices have been shown to operate on different levels of abstraction compared to experts [Tanaka and Taylor 1991] and may ignore features that are apparent to experts [Rosch et al. 1976]. In the context of the categorization and representation of

physics problems, Chi et al. find that novices have different mental models than experts and may categorize problems using surface features rather than deep structure [Chi et al. 1981]. While prior knowledge guides expert reasoning, novices' reasoning may be decontextualized [Shafto and Coley 2003]. This means that it may be more difficult to obtain novice interpretations that match an expert's understanding in areas that require prior knowledge, yet it also provides the opportunity to gain insights on how laypersons reason about certain topics [André et al. 2014]. This theoretical basis informs our decision to choose a decomposition approach, in which the goal mining task is decontextualized into micro tasks that do not require expert knowledge about privacy goals and vulnerabilities.

Task design plays an important role in obtaining high quality annotations from crowd workers. Tasks are usually decomposed into smaller micro tasks for crowdsourcing [Allahbakhsh et al. 2013]. Micro tasks can be assigned to multiple workers and individual annotations can be aggregated with different approaches [Hung et al. 2013] to obtain annotation reliability based on consensus. Reducing ambiguity and complexity of individual tasks and their instructions further improves result quality [Quinn and Bederson 2011]. Cheating can be discouraged by ensuring that cheating would require as much effort as completing the task honestly [Allahbakhsh et al. 2013]. Willett et al. further suggest to provide meaningful examples and feature-oriented prompts, to use detailed questions to focus the crowd workers' attention, and use highlighting and pre-annotations where appropriate [Willett et al. 2012].

A difficult problem is the aggregation of multiple worker annotations, as well as the recombination of answers obtained from multiple micro tasks into a coherent interpretation. Statistics can be used to remove outliers, but workers can also help in validating aggregated results [Sabou et al. 2012]. Hung et al. compared multiple result aggregation strategies and distinguished between non-iterative approaches, in which aggregation occurs after all annotations have been collected, and iterative approaches, in which the analysis of annotations influences the issuance of additional tasks, i.e., new tasks are issued to address weaknesses in previously collected task data [Hung et al. 2013]. They find that expectation maximization and supervised learning from multiple experts achieve the highest accuracy and are robust against cheaters. Zhang et al. propose a positive label frequency threshold approach [Zhang et al. 2013], which is similar to majority voting but takes into account that labels may be noisy and imbalanced due to certain response biases. Verroios and Bernstein propose Context Trees to recursively combine local summaries into a global interpretation of complex data [Verroios and Berstein 2014]. CrowdForge is a general-purpose framework for crowdsourcing complex and dependent tasks with a map-reduce approach, in which workers can decide how to subdivide a task [Kittur et al. 2011]. Result aggregation can be automated or partially performed by workers.

3. TASK RE-COMPOSITION FRAMEWORK

We now introduce our task re-composition framework, which combines human text interpretation with automated natural language processing to *re-compose* crowd worker micro task data into partial goal specifications based on the Eddy specification language [Breaux et al. 2014]. We present an example specification in Eddy's SQL-like syntax: the P indicates a permission, which is followed by a canonical description of the action `COLLECT`, the information type `email-contacts`,

the source keyword `FROM` and the actor from whom the information is collected (`zynga-users`), and finally the purpose keyword `FOR` followed by the purpose:

```
P COLLECT email-contacts FROM zynga-users FOR anything
```

These specifications are formalized in Description Logic, which is used to detect conflicts in policies [Breaux et al. 2013]. In this paper, we aim to recompose micro-task data to create a partial goal specification that includes the *action* and the *information type*, whereas we leave the *actor* and *purpose* for future work. We chose Eddy as our target language for expressing privacy goals, because it provides techniques for formally tracing data flows across privacy policies [Breaux et al. 2015]. However, manual extraction of the prerequisite privacy goals from policies to enable this traceability across thousands of policies is cost-prohibitive. The approach described herein aims to address this limitation

Figure 1 provides an overview of our hybrid framework that consists of two kinds of manual tasks (square boxes): tasks performed by an analyst, once (white boxes), or tasks performed by the crowd workers (red boxes); automated steps performed by tools (circles) and a reusable lexicon (parallelogram). The arrows point in the direction of data flows, e.g., illustrating where crowd worker annotations are sent to automated tasks; the solid vs. dotted lines signify separate but overlapping flows. We now discuss each step in more detail.

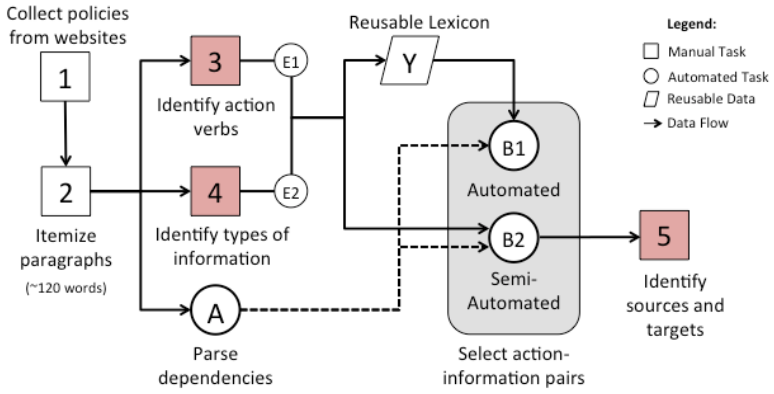

Fig. 1. Task re-composition workflow; red boxes represent crowd worker tasks.

During steps 1 and 2, the analyst prepares the input text to the NLP tools used in steps Y, B1 and B2 and the crowd worker platform, in this case Amazon Mechanical Turk (AMT), which is used in steps 3 and 4. These steps are performed manually by an analyst, once for each policy at present, because they require relatively little time (a few minutes per policy). For step 1, the input text begins as a text file, which may be extracted from a HTML or PDF document. For step 2, the analyst itemizes the text into paragraphs, averaging 90-120 words, that can be annotated in less than one minute by crowd workers, while ensuring that each paragraph's context remains undivided. For example, the analyst ensures that lists are not separated across tasks, and that anaphoric references, such as "it" or "this," are contained in the same paragraph as the noun phrases to which they refer. This invariant can lead to paragraphs that exceed 120 words, which is balanced by smaller 50-60 word paragraphs. The 120-word average limit determines the average time required by

one worker to annotate a paragraph, which we set to 60 seconds. This average time provides workers small, but frequent micro breaks between tasks and it allows workers frequent opportunities to stop annotating text whenever they feel fatigue or boredom. Because the tasks are small and independent, workers can stop at any time and workers need not complete all of the tasks for a single policy: subsequent workers can be given tasks that continue where previous workers stopped working. The small tasks also allow us to better distribute the risk of low-performing crowd workers and the associated costs.

3.1 Crowd Worker Micro Tasks

Steps 3 and 4 are crowd worker micro tasks that ask workers to annotate phrases in one of two ways: for step 3, workers are asked to label action verbs that describe information collection, use, transfer or retention, as shown in Figure 2. Following simple instructions, workers see the ~120-word paragraph and are tasked to select and annotate relevant phrases using their mouse and keyboard. The annotated phrases are color coded to correspond to the label selected by the worker. The micro task for step 4 is similar, except that instead of distinguishing among four kinds of actions, workers are asked to identify noun phrases that correspond to any kind of information. In both steps 3 and 4, the results are captured and recorded as part of an AMT batch result, wherein we asked five workers to annotate each paragraph. This number of workers was determined in prior work, which showed worker agreement for 2/5 workers correlates with high precision and recall for these tasks [Breaux and Schaub 2014].

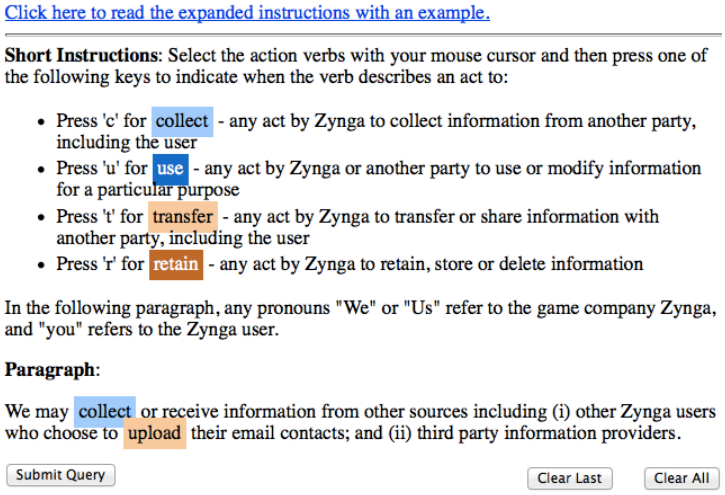

Fig. 2. Crowd worker micro task to annotate information actions.

As shown in Figure 1, the results from steps 3 and 4 are combined with a dependency parse of the paragraphs to select action-information pairs in steps A, B1, B2, which we now discuss.

3.2 Dependency Parsing and Pair Selection

In step A, we apply typed dependency parsing to the individual sentences from the micro task text input using the Stanford dependency parser [Marne et al. 2006]. Typed dependencies are binary relations between a first term, called the *governor*, and a second term, called the *dependent*. We present an example sentence with the

corresponding collapsed, CC-processed dependencies (collapsed dependencies with propagation of conjunct dependencies) for each word in the sentence in Figure 3. Commonly found dependency types include *nsubj*, which is the nominal subject of the sentence, and *dobj* or direct object of a verb phrase. One advantage of dependency parsing is that the parser splits phrases along conjunctions and it links modifiers to nouns. However, natural language ambiguity can lead to errors in parsing. For example, Figure 3 presents three dependences *dobj(collect, providers), dobj(collect, information),* and *dobj(upload, contacts).*

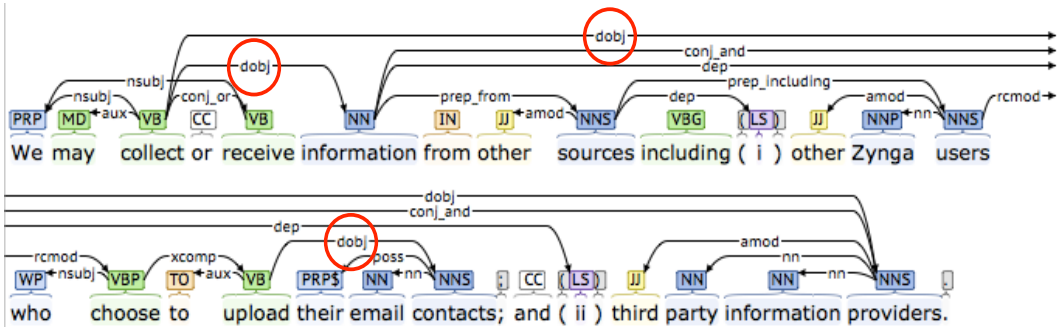

Fig. 3. Stanford dependency parse of micro task input text

The first dependency *dobj(collect, providers)* is incorrect: the sentence author likely did not mean that the website "collects third party information providers;" rather, the providers are a second example of "from whom" that information is collected. Thus, we assume some degree of inaccuracy produced by the typed dependency parser. However, the second two dependencies are correct and they indicate prospective goals about which we can ask additional questions: "from whom is information collected" (a collection goal), and "by whom are contacts uploaded" (a transfer goal). Our approach to select action-information type pairs is limited by the accuracy of the Stanford Parser.

We propose two different approaches, denoted by steps B1 and B2, to select the action-information type pairs using the typed dependencies. The typed dependencies are combined with crowd worker annotations in step B2, wherein we perform action–information pair selection to identify actions (typically verbs) that should be paired with information types (typically noun phrases). Step B2, is a semi-automated approach that requires manual annotations for the actions and information types, which are obtained from the crowd workers.

In order to automate this process of obtaining the action and information types, we propose an alternate approach in step B1, which is a fully automated approach. In this step, we use the action and information type lexicon, to identify actions and information types in the policy statements using a simple keyword match between the lexicon and policy terms. We combine these identified actions and information types in each statement with the typed dependencies of the statement to determine if the action and information type are linked by a typed dependency or not. If linked, the corresponding action-information type pair is selected as a candidate partial goal specification. We use two general strategies for both steps B1 and B2 for linking action-information pairs: (1) we first identify *direct* dependencies, in which both the governor and dependent were separately annotated by either the lexicon for B1 or by crowd workers for B2 in the action and information type tasks; and (2) we identify *indirect* dependencies that consist of two typed dependencies, each one containing

one lexicon- or worker- annotated term and sharing a third term, which may not have been annotated. We only consider terms that have been annotated by the lexicon in step B1 or by two or more crowd workers in step B2 based on prior work that shows 2/5 workers produce high precision and recall for these tasks [Breaux and Schaub 2014]. In Figure 3, for example, *dobj(upload, contacts)* is a direct dependency, if "upload" was annotated by two or more workers in the action task, and "contacts" was annotated by two or more workers in the information type task. In addition, in Figure 3, *dobj(collect, information)* and *cc_or(collect, receive)* comprise an indirect dependency that links *receive* to *information* via the *cc_or* typed dependency for the English conjunction "or". In our evaluation, we are interested in identifying which dependency types are high confidence, meaning, they maximize true positives and minimize false positives.

Next, we introduce the lexicon as a means to collect and reuse knowledge about annotated actions and information types to improve recall (missing true positives in step B2) and to develop the fully automated approach for step B1.

3.3 Reusable Lexicon and Entity Extraction

Lexicons are used to bootstrap requirements analysis by re-using terms frequently seen in particular domains. In our work, we build the lexicon using crowd worker annotations from steps 3 and 4 in Figure 1 for 30 privacy policies to attempt fully automated goal finding. The lexicon is constructed from action and information type entities, which are unique textual descriptions needed to identify recurring instances of the same concept. For instance, the entities in the lexicon should enable us to resolve synonyms, plurals and singular forms of information types (e.g., "email address" is basically the same concept as "email addresses"). In steps E1 and E2, we apply an entity extraction technique on the annotated verb and noun phrases provided by the crowd workers to extract the individual entities (information types) from the annotated phrases. These phrases may consist of ambiguous lists and clauses that obfuscate the unique entities. The entity extractor was first evaluated on 3,850 crowd worker information type annotations [Bhatia and Breaux 2015]. In Section 4.4, we present an extended evaluation on 7,682 annotations from 30 policies and results of applying the acquired lexicon to the re-composition framework.

The entity extractor workflow is presented in Figure 4. The extractor first tests whether a worker annotation is a list (i.e., it contains a common list delimiter, such as a comma, semi-colon or POS-tagged English conjunction CC). If an annotation does not contain a list delimiter, then we test whether the annotation describes a single entity by checking the annotation's POS tag sequence against a well-known regular expression NP + CL that matches a noun phrase (NP) followed by a clause (CL) expressed as standard POS tags[2] as follows [Justeson and Katz 1995]:

$$NP=((JJ\,|\,RB\,|\,VBG\,|\,VBD\,|\,NN\backslash S?\,|\,NN\backslash S?\backslash sPOS)\backslash s)*(NN\backslash S?)$$

$$CL=\ (\backslash s(IN\,|\,PRP\,|\,TO\,|\,VBG\,|\,VBN\,|\,WDT\,|\,WP)\backslash s.*)?$$

[2] *IN: Preposition or subordinating conjunction, JJ: Adjective, NN: Noun, POS: Possessive ending, PRP: Preposition, TO: to, RB: Adverb, VBD: Verb, past tense: VBG Verb, gerund or present participle, VBN Verb, past participle.*

Based on our analysis of 30 policies, 71.4% of the worker supplied information type annotations describe single entities, and the remaining 28.6% describe lists. For lists, the extractor checks whether the annotation describes a modified noun, which comprises 1.9% of annotations. This case includes lists of conjoint adjectives followed by a noun (e.g., "aggregate, statistical information"), as well as disjoint lists (e.g., "geographic and demographic information"). Disjoint lists are split to distribute the modifiers separately across the nouns (e.g., to yield "geographic information" and "demographic information"). The remaining 26.7% of annotations are lists of noun phrases, which are split by delimiter. Each delimiter-separated noun phrase is checked against previously seen simple, non-obfuscated entities, called ground terms. In Figure 4, ground terms are automatically identified where the output boxes are colored blue. While the workflow is seemingly complex, it has been shown to be highly effective at extracting entities, as we discuss in Section 4.4.

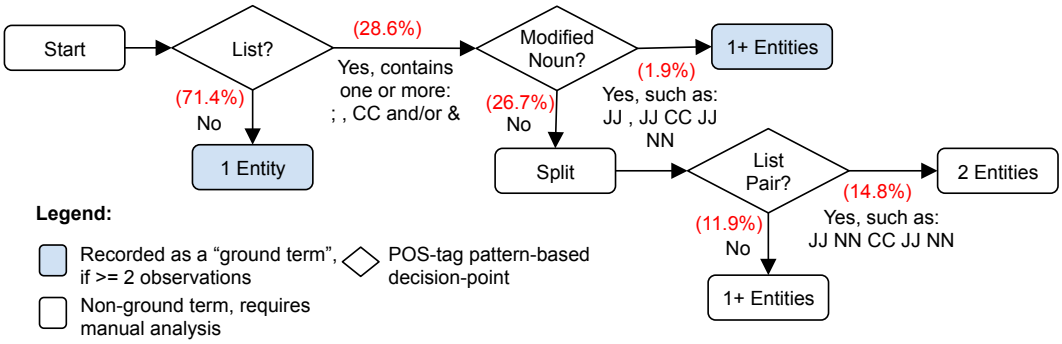

Fig. 4. Workflow for the information entity extractor

We next discuss the crowd worker pair validation task in step 5.

3.4 Validate Pairs and Identify Source and Target

In step 5, we take the selected action and information type pairs from step B2 and send these pairs to the crowd workers to ask whether the information action and information type are valid pairs. If true, we also ask crowd workers to identify the actors who send, receive and use the information based on the coded action type. This validation task helps us to remove the false positive action-information type pairs produced by step B2, because unlike the crowd workers who understand context, the lexicon indiscriminately identifies all candidate pairs based on keyword matches. Figure 5 presents the task interface for step 5, in which workers select the action modality ("permits" or "prohibits"), the action category, and then they complete the source and target questions using radio buttons or free-response text boxes. If the worker selects collection from the drop-down list, the questions ask "from whom," whereas selecting transfer from the drop-down list asks "to whom." For use and retention, we ask only "by whom" is the information used or retained.

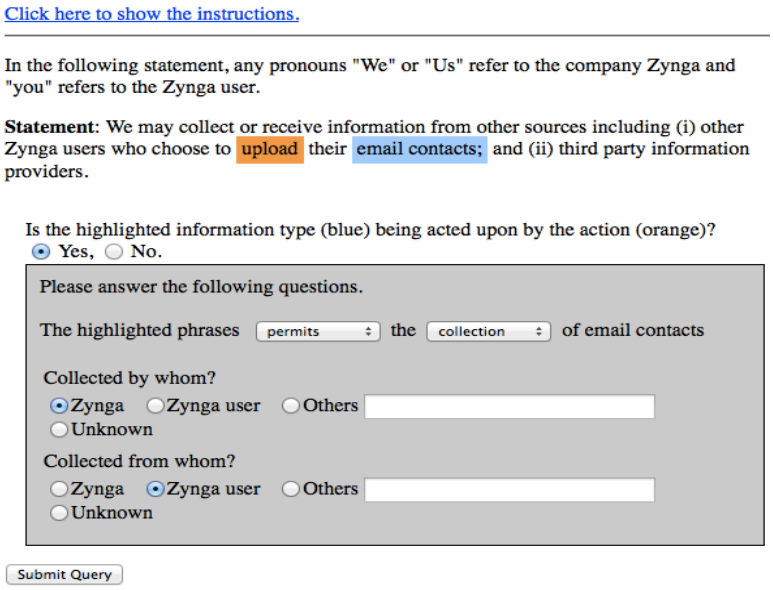

Fig. 5. Crowd worker micro task to validate action-information type pairs

Our framework makes use of crowd worker annotations, to identify the actions and information types in the privacy policy statements, which are linked using the typed dependencies to select the action-information type pairs for each statement. The action and information type annotations are used to build the action and information type lexicon respectively, which are reused to identify missing crowd worker annotations, and to attempt to fully automate the crowd worker annotation process. We next present the results of our framework evaluation.

4. EVALUATION AND RESULTS

The task re-composition framework presented in Section 3 combines human annotation with natural language dependency parsing and a reusable lexicon to identify action-information type pairs that comprise partial specifications of data processing.

We evaluated the framework by answering the following research questions:

RQ1. How do crowd workers compare with expert annotators in performing micro tasks?

RQ2. How well do typed dependencies combined with crowd worker annotations predict the pairs needed to express partial goal specifications?

RQ3. How well does the lexicon improve identification of missing annotations or pairs?

RQ4. How well does lexical reuse increase with each new policy analyzed?

RQ5. How well do crowd workers identify false positives in a validation task?

Research question RQ1 evaluates steps 3 and 4 in the framework (see Figure 1) with respect to precision and recall using the expert annotations. This evaluation extends a prior evaluation of these two tasks that examined only a single policy

[Breaux and Schaub 2014]. Research question RQ2 evaluates the typed dependency step A and pair selection step B2 against the expert pairs, while RQ3 separately evaluates pair selection steps B1 using the crowd worker annotations and lexicon against the expert pairs. Research question RQ4 evaluates the lexicon independently to assess how it scales over time. Finally, research question RQ5 evaluates step 5 and the ability of the crowd workers to identify false pairs against the expert pairs.

To evaluate our hybrid framework and to answer the research questions, we selected five privacy policies that the first two authors (the experts) analyzed as part of this case study, which we refer to as *expert annotations* and *expert pairs* when combined into a partial goal specification:

- AOL Advertising, last updated 4 May 2011
- Facebook API Developer Guidelines, revised 28 June 2013
- Flurry Privacy Policy, updated 9 July 2013
- Waze Privacy Policy, modified 30 May 2013
- Zynga Privacy Policy, last updated 30 Sep 2011

These policies were selected because they were used in two prior case studies to express privacy goals formally in the Eddy language based on Description Logic [Breaux et al. 2014; Breaux et al. 2015]. The policies correspond to different stakeholders in a software composition: the AOL and Flurry policies govern advertising services used by a game provider (Zygna) and a navigation application (Waze). The Facebook policy governs a platform that both Zynga and Waze use for user identification services, when users log in to their applications using their Facebook accounts. Thus, these policies cover two popular data flows and the policy language in each of these policies varies by the role of the covered services (ad services, identity provider, and first-party app developers).

The expert data set was created by two analysts (the first and second authors) by extending the annotations from a prior case study [Breaux et al. 2014]. In this prior study, on average, the first analyst expended 1.09 minutes per statement extracting requirements, whereas the second analyst expended 2.21 minutes per statement [Breaux et al. 2014]. For the new data set, the analysts spent on average 1.9 minutes each per statement to review the previous annotations and extend the dataset. The expert data set serves as the "ground truth" by which we compute precision and recall as measures of the automated steps B1 and B2 shown in Figures 1, above. For all precision and recall calculations, the expert data set contains the sum of true positives and false negatives.

We now discuss our empirical results with respect to each research question.

4.1 Crowd Worker Micro Task Results

We solicited five workers per micro task to identify the actions and information types. We recruited US residents as workers on AMT, who had at least a 95% approval rating for over 5,000 tasks. We paid workers $0.15 per task for actions and $0.12 per task for information types to keep the hourly wage close to $8-10 per hour. We allowed up to five minutes to complete each task. Results were accepted or rejected within 24 hours. For the action identification task, the workers required 72 seconds on average to complete a single task, which resulted in an average hourly

rate of $8.40. On average, workers required 61 seconds per information type task, with an average hourly rate of $6.30.

Table I presents the total cost incurred for the information action and information type identification micro tasks for all policies, including: the total number of tasks (*Tasks*) in each policy; Amazon charges of 10% (*AMT fees*), and *Total Cost, consisting of worker payments and AMT fees*.

Table I. Cost to Crowdsource the Micro Tasks

Policy	Actions Micro Task			Info. Types Micro Task		
	Tasks	AMT fees	Total Cost	Tasks	AMT fees	Total Cost
Waze	34	$2.55	$28.05	34	$2.04	$22.44
Zynga	32	$2.40	$26.40	32	$1.92	$21.12
Flurry	33	$2.48	$27.23	33	$1.98	$21.78
FB	32	$2.40	$26.40	32	$1.92	$21.12
AOL	18	$1.35	$14.85	18	$1.08	$11.88

Table II presents the number of annotations acquired from steps 3 and 4: for each policy, we present the total number of sentences in the policy, the total number of sentences with annotated actions only, with annotated information types only, with both an annotated action and information type, and finally the overall total number of annotated actions and information types. For sentences with only the annotated actions or information types and not both, these sentences would not yield an action-information type pair based on an expert analysis of the text.

Table II. Summary of Micro Task Annotations

Policy	Total Sentences	Sentences with:			Annotations	
		Only Actions	Only Info Types	Both	Actions	Info Types
Waze	117	5	36	56	117	146
Zynga	97	4	28	52	103	125
Flurry	135	22	32	49	106	111
FB	136	15	25	57	129	166
AOL	76	6	6	50	96	87

Table III presents the precision and recall for both actions and information types as compared to the expert annotations. On average, workers were able to identify the actions and information types with high recall of 0.84 and 0.92, respectively and average precision of 0.87 and 0.83, respectively. Notable in Table III, the Flurry policy includes nomenclature specific to the advertising industry that crowd workers are likely unfamiliar with, which may explain the lower precision and recall for that policy as compared to the other policies.

Table III. Crowd-Sourced Action and Information Type Annotations Compared to Expert

Policy	Actions		Information Types	
	Precision	Recall	Precision	Recall
Waze	0.88	0.83	0.62	0.91
Zynga	0.91	0.79	0.95	0.98
Flurry	0.73	0.64	0.97	0.84
FB	0.98	0.96	0.88	0.90
AOL	0.86	0.98	0.71	0.95
Average	0.87	0.84	0.83	0.92

4.2 Dependency Parse and Pair Selection Results

We now present the dependencies parser results and results of our techniques for selecting action-information type pairs.

Table IV presents results from a naïve approach to produce typed dependencies from the five policies to illustrate the scope of the pair selection challenge. This includes the number of unfiltered dependencies per policy (*Total Dependencies*); the subset of the total in which the governor or dependent are a verb and noun pair (*Dependencies w/ Verbs & Nouns*); the three most common dependency types found in the direct selection method described in 3.2, which are *dobj* (direct object of a verb phrase), *nsubjpass* (syntactic subject of a passive clause) and *vmod* (verb heading a phrase); and the number of pairs identified in the expert analysis, which represents our evaluation target. As can be seen from Table IV, the space of dependencies is quite large and, assuming perfect recall, the precision of a naïve approach to pair selection would be very low.

Table IV. Naïve Approach to Identify Relevant Pairs – Parser

Policy	Total Dependencies	Dependencies w/ Verbs & Nouns	dobj, nsubjpass, vmod	Expert Pairs
Waze	3286	794	365	101
Zynga	2758	655	352	93
Flurry	3268	845	398	81
FB	3389	765	339	91
AOL	1720	452	216	81

In Table V, we present a slightly more informed approach to identify action and information type pairs using typed dependencies and lexicon (B1 in Figure 1). The column *Expert Pairs* lists the total number of action and information type pairs identified by the experts, manually. The column *Lexicon and Parser Pairs* lists the total number of pairs automatically obtained by pairing actions and information types from the lexicons that share a *direct* or *indirect* dependency based on the parser output. The columns *Precision* and *Recall* are computed by comparing the *Lexical and Parser Pairs* to the *Expert Pairs*, which serves as the ground truth. The lexicon-based approach was able to identify the action-information type pairs with an average recall of 0.80, however, the average precision was very low at 0.20. The large number of false positives obtained using the lexicon can be attributed to the fact that at present the lexicon does not have the ability to disambiguate the meaning of a term in the given context, and thus identifies all instances of a term in a statement.

Table V. Naïve Approach to Identify Relevant Pairs - Parser and Lexicon

Policy	Expert Pairs	Lexicon and Parser Pairs	Precision	Recall
Waze	101	360	0.22	0.77
Zynga	93	424	0.19	0.86
Flurry	81	432	0.15	0.79
FB	91	306	0.22	0.75
AOL	81	229	0.22	0.79

From Table IV and V, we see that semantic dependencies alone, even direct dependencies without human guidance, produce a large number of false positives compared to the evaluation target. In addition, while the lexicon contains

terminology from prior worker annotations, it lacks the workers' direction in reducing the dependencies to within a reasonable reach of the evaluation target. To inform our approach, we analyzed the direct and indirect dependencies to determine the most frequent dependency patterns found in the re-combinations and how often they lead to true or false positives. We found three direct dependency patterns and five indirect dependency patterns that constitute 71.81% of the total true positive re-combinations. We describe these patterns in Table VI as follows: the pattern name, the typed dependency sequence, the frequency of the pattern across all five policies, and the number of true and false positive action-information type pairs for each pattern measured against the expert pairs.

Table VI. Typed Dependency Patterns

	Pattern Name	Typed Dependency Sequence	Frequency	True Positive	False Positive
Direct	Direct Object	dobj	195	188	7
	Passive nominal subject	nsubjpass	34	32	2
	Verbal modifier	vmod	24	22	2
Indirect	Conjunction and with direct object	conj_and , dobj	25	15	10
	Conjunction or with direct object	conj_or, dobj	17	12	5
	Passive nominal subject with list.	nsubjpass, prep_such as	1	1	0
	Direct object with verbal modifier	dobj, vmod	13	2	11
	Direct object with preposition	dobj, prep_*	20	10	10

The three direct dependency patterns (direct object, passive nominal subject and verbal modifier) in Table VI on average constitute 59.1%, 10.3% and 7.3%, respectively, of the direct dependency re-compositions across all five policies for the hybrid approach. These three patterns led to true positives in 99.6% of the instances studied. The only instance where the direct object pattern yields an incorrect result was "You must immediately revoke an end-advertiser's access to your app upon our request." (from Facebook privacy policy) In this sentence, *revoke* is annotated as an information action and *access* is annotated as an information type by the workers. The pair (*revoke*-access) is linked by a direct object dependency, which is a true dependency yet a false positive because "access" is not an information type.

The five indirect dependency patterns describe 41.1% of the total indirect dependency re-compositions in the hybrid approach action and information type pairs. On average, the direct dependency patterns led to true positives in 87.9% and indirect dependency patterns led to true positives in only 44.3% of the instances.

As observed from Table IV and V, there is no simple approach to using the parser to identify the action-information type pairs. By adding our crowd worker annotations for both the actions and information types, however, we identified a set of high confidence pairs that consist of the direct and indirect pairs defined in Section

3.2. Ideally, these pairs will contain all true positives and minimal false positives, and omit minimal false negatives. In Table VII, we present our baseline measure (Total Annotated Pairs), which is the number of all possible pairs, which assumes naively that every annotated information action is crossed with every information type that occurs in the same sentence, followed by the total number of high confidence pairs based on dependency parsing and worker annotations, and the total number of expert pairs. The hybrid approach greatly reduces the number of pairs as compared to the naïve approaches presented in Tables IV and V.

Table VII. Action-Information Type Pairs from Hybrid Approach

Policy	Total Annotated Pairs, Possible	High Confidence Pairs	Expert Pairs	Precision	Recall
Waze	379	107	101	0.73	0.77
Zynga	467	120	93	0.64	0.83
Flurry	237	71	81	0.79	0.69
FB	301	111	91	0.75	0.91
AOL	239	106	81	0.74	0.96

In Table VII, the Flurry policy has the lowest precision and recall among all analyzed policies. This is because workers annotated both the information action and information type in only 36.3% of the sentences in the Flurry policy (see Table II), whereas in other policies workers annotated 52.3% on average. The actions in the Flurry policy that were not identified by the workers were context-sensitive – e.g., "get back" (a colloquialism), and "export" and "request" (both software functions), to name a few – which were also different from the action words frequently found in other policies. Thus, the workers biased by terminology commonly found in other policies may have not expected and thus missed these terms.

Our analysis of the remaining 143 false positive pairs after the expert analysis shows that 14/143 pairs contain an action that was part of a data purpose. For example, in the sentence "We use personal information to *create* new *services*", the action "create" marks the beginning of the purpose for which the information type personal information is being used. We observed that 12/143 pairs were pairs where a technology was being used to perform an information action. For example "This information is *collected* by the use of *log-files*." In this case, the log-files are a container for information and a technology. Manually excluding such pairs from our analysis would improve average precision from 0.73 to 0.78, which offers promise for future work.

In addition, we manually analyzed the 75 false negative sentences from all five policies in which the action and information type pairs were identified by the experts, but were missed by our crowd workers. Our analysis shows that out of the 75 sentences, the workers did not annotate an information action in 37/75 of these sentences; in 20/75 of these sentences the workers did not annotate an information type; and in 5/75, the workers did not annotate both the action and the information type. In Section 4.3, we discuss how we make use of the reusable lexicon to identify these missing annotations and reduce the number of false negatives.

In the remaining 13/75 sentences, the workers had identified the information action and the information type, but the parser could not determine a direct or

indirect dependency between the information action and information type in the pairs, thus, they were not included in our high confidence pairs. On further inspection, we found that this was because of incomplete worker annotations. For example in the sentence, "*You can retrieve recommendations created for a particular End User by passing the device identifier of the End User*" the workers annotated the information action *retrieved* but missed its corresponding information type (*recommendations*). Instead, they annotated the information type *device identifier*, but missed its corresponding information action (*passing*). The annotated information action and type pair (*retrieved-device identifier*) is not linked by a direct or indirect dependency relationship and was therefore excluded from the high confidence pairs.

In the next section, we discuss the reusable lexicon's impact on identifying missing actions and information types.

4.3 Impact of Reusable Lexicon on Pair Selection

We now present the results of the reusable lexicon. We built the lexicon from 30 policies spanning five domains: employment, news, social networking, shopping, and telecommunications. The five policies listed in Section 4 were not part of the selected policies. The entity extractor successfully extracted entities from 97.8% of crowd worker annotations. In Table VIII, we present the number of actions and information types that were missing from the crowd worker annotations and identified using the lexicon, and the corresponding number of missing high confidence direct and indirect pairs that result from applying the lexicon to each of the five policies that are used for evaluation.

Table IX presents the number of false negative pairs produced from worker annotations reported in Section 4.2, the number of true positive pairs identified using the high confidence pairs from the lexicon reuse reported in Table VIII, and the precision and recall without the lexicon reported in Table VII, and the precision and recall with the lexicon. The results in Tables VIII and IX were computed using all the terms in the action lexicon and information type lexicon.

The lexicon-produced high confidence pairs identified 37.34% of the pairs that were FNs from the worker annotations and improves the average recall by 8.8% to 0.90. However, the lexicon also significantly reduces the average precision by 31% to 0.50 based on the expert pairs. But as it has been previously noted by Berry et al., it is difficult to achieve both high precision and high recall with NLP techniques for requirements engineering, and a NLP tool for requirements engineering should be tuned to favor recall over precision because errors of commission are generally easier to correct than errors of omission [Berry et al. 2012]. We therefore aim at minimizing the number of false negatives, even if that means accepting a few false positives.

Table VIII. Results for Reusable Lexicon

Policy	New Actions, Missed	New Information Types, Missed	New High Confidence Pairs, Missed
Waze	116	17	58
Zynga	165	19	74
Flurry	88	99	78
FB	81	79	51
AOL	20	36	26

Table IX. Impact of Lexical Reuse on Precision and Recall

Policy	False Negative Pairs	True Positive Pairs	Precision w/o Lexicon	Precision w/ Lexicon	Recall w/o Lexicon	Recall w/ Lexicon
Waze	23	6	0.73	0.51	0.77	0.83
Zynga	16	6	0.64	0.43	0.83	0.92
Flurry	25	12	0.79	0.46	0.69	0.84
FB	8	3	0.75	0.52	0.91	0.95
AOL	3	1	0.74	0.60	0.96	0.98

When using all the terms from the action and information type lexicons, precision drops by 31% for an 8.8% increase in the recall over the hybrid pair results in Table VII. This decrease in precision is due to the effect of context in terminological reuse. Phrases, such as "send" or "receive" may indicate information collection and transfer in one context, but be used to describe non-information transactions in another context. To find the optimal subset of the lexicons that leads to an increase in recall without a steep decrease in the precision, we conducted an experiment based on lexicon partitions portioned from increasing increments of 10%. In Table X, we present the Precision and Recall for different lexicon partitions. The column, *Action Lexicon* shows the partition of the action lexicon that was used for the respective experiment. In Table X, *x% Action Lexicon* means that, top x% of the terms in the action lexicon were used for the analysis. Similarly, *X% Info. Type Lexicon* means that the top x percent of the terms in the information type lexicon were used for the analysis.

Table X. Impact of Lexical Reuse on Precision and Recall

Action Lexicon	10% Info. Type Lexicon		50% Info. Type Lexicon		100% Info. Type Lexicon	
	Precision	Recall	Precision	Recall	Precision	Recall
10% Action Lexicon	0.63	0.88	0.61	0.89	0.60	0.89
50% Action Lexicon	0.56	0.89	0.54	0.90	0.53	0.90
100% Action Lexicon	0.53	0.90	0.52	0.90	0.51	0.91

From our experiments with the lexicon partitions, we conclude that the precision decreases and recall increases as the partition size of the lexicon for the experiment increases. Further, the decrease in precision is greater than the increase in recall.

The precision drops by 14.3% and the recall increases by 5.4% over the hybrid pair results in Table VII when we use the top 10% terms in the action and information type lexicons, The precision further decreases as we increase the partition sizes and the precision drops by 31% when the entire action and information type lexicons are used, for an increase of 8.8% increase in recall.

Even after lexical reuse, some information actions from the expert annotations could not be identified using the hybrid approach and lexical reuse. These actions include "based on," "get back," "complete" (a user profile), and "be visible," which are context-sensitive or require rich interpretations, such as multiple inferences or tacit knowledge (e.g., "be visible" suggests that others can see the information that has been visible and this inference constitutes a form of information transfer). The lexicon is missing some information types, which include domain-specific information not present in the lexicon, for example, "customized audience" and "identifiable-route." Missed information types also include anaphora, such as "it," that refer to an information type in the prior sentence, which was identified by the experts, but not by the workers.

The first and third authors evaluated the lexicon to determine the scale of false positives that the lexicon can introduce when used without worker annotations. These two authors analyzed the Waze and Zynga policies to identify those instances of actions and information types that appear in the lexicon, but were not annotated by the workers (i.e., to find possible false negatives). They identified 909 actions and 450 information types in the two policies, among which only 15% of the actions and 12.2% of the information types were false negatives. From this analysis, we conclude that worker ability to distinguish between true positives and false positives is an improvement over the lexicon alone, and the lexicon alone could greatly inflate the number of false positives, if used without worker annotations.

In summary, the low precision due to the lexicon can be attributed, in part, to the ambiguity of terms and the role of context in determining when data processing events take place, and to the noise in worker responses. Information actions that are ambiguous include "assist," "solicit," "permit," and "allow," among others. Terms that workers annotated as information types that should be excluded include "third parties," "campaign" and "network." In the case of "campaign" and "network," these are activities and technologies that imply some type of information, but are not themselves the implied information type. We also observed that false negatives in the worker data include words that occur less frequently across policies, including actions, such as "permit" and "export," and information types, such as "payment," "ads." Thus, limiting the lexicon to the most frequent words and phrases, will in turn hinder the ability of the lexicon to identify false negatives.

4.4 Results of Scaling Reusable Lexicon

We examined the extent to which the lexicon can predict actions and information types in additional privacy policies. This analysis shows that privacy policies have unique entities that are not shared across policies. Figure 6 presents the saturation (sat.) of information type entities for the same 30 policies: at any point along the x-axis, we observe the percent reuse of information types in a policy N based on the last N-1 policies previously seen. This result is based on 100 pseudo-random permutations of the orders of the 30 annotated policies. We observe that near 14-15

policies, the average maximum threshold for saturation of 77% is achieved, meaning, every new policy contributes a sufficient number of unique terms to the lexicon that 23% of the new policy would not appear in any previously seen policy in the best case, and 71% of the policy terms would be new in the worst case. At present, this observation suggests that the lexicon cannot entirely replace crowd workers, because there will always be new terms that the lexicon has never encountered.

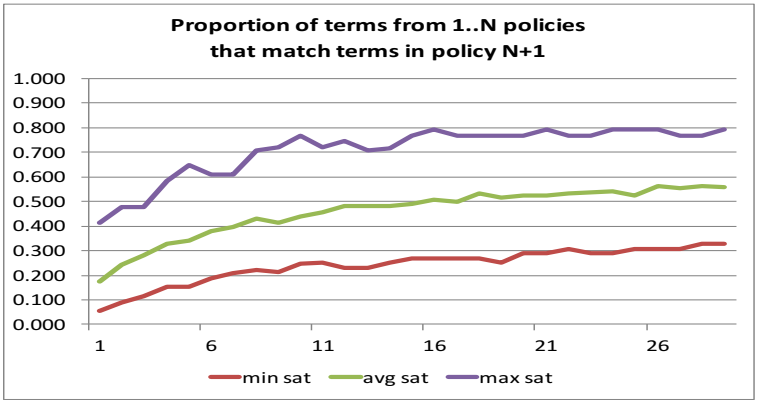

Fig. 6. Saturation of information types in lexicon: we observe % reuse of information types described in N-1 policies for each N'th policy along x-axis.

We further analyzed the action verbs from the same 30 policies and found 377 unique verbs identified by crowd workers. Only a small subset of these verbs dominate the results, with 10% of action verbs describing 75% of the annotations (see Fig. 7, which shows the number of annotations per verb on the y-axis in logarithmic scale, and each indexed verb along the x-axis). There is ambiguity, however, with 28% of verbs coded by two or more actions (collect, use, transfer and retain) and 5% of verbs coded as sharing-ambiguous, meaning they were coded as both collect and transfer by two or more workers. For these verbs, it may be difficult for crowd workers to determine from the text who is providing and who is receiving the relevant information. Finally, some of the verbs were also used to describe use-related purposes, which is one source of reduced precision discussed in Section 4.3

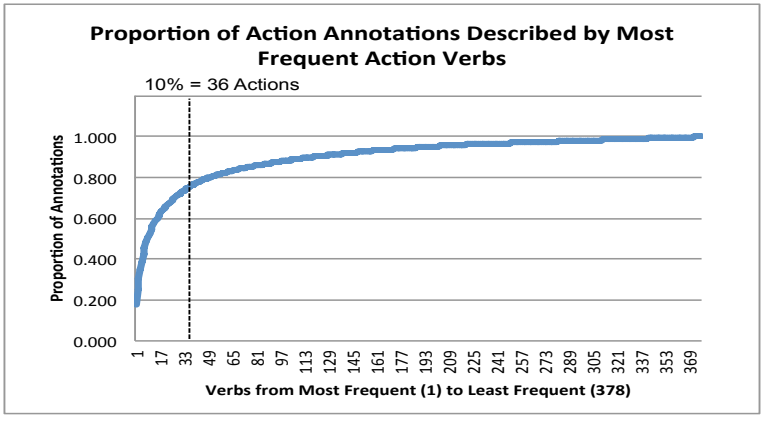

Fig. 7. Number of annotations per verb along the y-axis (log scale), and each unique verb of 380 verbs along x-axis; 10% of verbs covered 75% of annotations

4.5 Validation Task Result

The results from Section 4.2 show that the high confidence pairs from our hybrid approach contain some number of false positives (see step A and B2 in Figure 1, and Tables VII and VIII). One objective of step 5 in our framework (see Figure 1) is to identify these false positives and remove them from the results to achieve higher average precision. In this validation task, we ask workers whether the action and information type pair from the high confidence pairs is a valid pair (true positive), or whether it is an invalid pair (false positive). If a valid pair, then we ask crowd workers to identify the modality and the actors who send, receive, and use the information based on action labels provided by the previous crowd workers in step 4.

We solicited five workers per task to identify the valid information actions and information type pairs. We recruited US residents as workers on AMT, who had at least a 95% approval rating for over 5,000 tasks. We paid workers $0.12 per classification task. We allowed up to five minutes to complete the task. Results were accepted or rejected within 24 hours. The workers completed each task in 39.6 seconds on average, resulting in an average hourly rate of $10.95.

In the analysis of the validation task, we mark a pair as false positive, if more workers annotated it as false positive than the number of workers who annotated it as true positive. Table XI presents the validation task results as follows: the number of high confidence pairs obtained using our hybrid approach (see results in Section 4.2 from step B2 in Figure 1); the number of false positive pairs identified by three or more workers, the number of false positive pairs identified by the experts; the number of ambiguous pairs, in which 2/5 and 3/5 workers yielded conflicting annotations; the precision without validation reported in Table VII; and the precision with validation from crowd workers. As shown in Table XI, the crowd workers greatly reduced the number of false positives produced by the direct and indirect dependency patterns.

Table XI Pairs Validation Result

Policy	High Confidence Pairs	False Pairs by Worker	False Pairs by Experts	Ambiguous Pairs	Precision w/o Validation	Precision w/ Validation
Waze	107	20	30	12	0.73	0.88
Zynga	120	44	43	13	0.64	0.94
Flurry	71	11	16	4	0.79	0.92
FB	111	27	28	20	0.75	0.91
AOL	106	32	28	17	0.74	0.99

5. THREATS TO VALIDITY

In this section we discuss the threats to construct, internal, and external validity for our framework, which makes use of crowdsourcing and NLP techniques.

5.1 Construct Validity

Construct validity describes whether what we proposed to measure is indeed what we measured [Yin 2009]. One concern in our framework is whether crowd workers accurately understood what constitutes an action and information type during the annotation task. To address this concern, we provide detailed instructions and a worked example in the micro task description to help crowd workers understand what kinds of phrases match our interpretation of action and information types (see

Section 3.1). Furthermore, we consider only annotations where two or more crowd workers agreed on the same annotation. We also compare these crowd worker annotations to expert annotations to measure the extent to which crowd workers agreed with the experts. On average, workers identified actions and information types with high average recall of 0.84 and 0.92 respectively, and high average precision of 0.87 and 0.83, respectively as shown in Table III. Sources of disagreement often arose when a given word or phrase is ambiguous. For example the action "*access*" can be annotated as collect, use or transfer, depending on who is performing the action (the end user, the company, or a third party).

5.2 Internal Validity

Internal validity is the extent to which observed causal relationships exist within the data and, particularly, whether the investigator's inferences about the data are valid [Yin 2009]. In our framework, our conclusions depend on the reliable performance of the Stanford Parser to identify typed dependencies. For example, the per-dependency accuracy of the Stanford Parser has been reported to be 80.3% [Marne et al. 2006]. Changes in the parser accuracy will affect overall performance. In addition, the types of policies that we studied could present potential confounds in the form of selection bias: policies with more or less technical information types or policies that describe new, previously unknown technologies, could be harder for workers to annotate, because they may not recognize the associated information types. The task of interpreting natural language text is subjective and even the experts could miss an annotation, or the experts could be influenced by the interpretations discovered by the crowd workers. But for the purpose of evaluating our approach, we use the annotations generated by the experts as ground truth. We therefore note that this expert dataset contains the true positives and false negatives for our evaluation purposes.

5.3 External Validity

External validity is the extent to which our approach generalizes to the population outside the sample used in the study [Yin 2009]. Based on our study of crowd worker interpretation of the privacy policy text (see results in Table III) and on the Stanford Parser identification of typed dependencies linking annotated words, we believe our approach could be used for other information-related legal texts. However, we hesitate to make this claim without conducting further studies with different legal text and in other domains. For complex laws, such as the Privacy Rule of the Health Insurance Portability and Accountability Act (HIPAA), there may be additional steps required to preprocess and convert the documents into micro-tasks for crowdsourcing, which we further discuss in Section 6 that follows.

6. DISCUSSION AND SUMMARY

We now discuss our hybridized re-composition approach in the light of our results and the potential for future work. The analyses conducted for the five research questions show complimentary results. With respect to RQ1, which compares the crowd worker performance to expert annotators, we find that untrained crowd workers can be used to elicit most of the actions and information types that were identified by the experts, which leads to high recall for the crowdsourcing action and information type micro tasks, when compared to the expert annotations. Moreover, we discovered that many false positives are due to natural ambiguities in the text and task description that are difficult to remove. A complementary finding that

answers RQ2, which asks about the performance of dependency parsing alone, suggests that context and tacit knowledge are required to identify relevant actions and information type pairs. The crowd worker annotations, which are reasonably low cost to acquire, can be used as guidance for selecting parser dependencies to identify a set of high confidence pairs. Our results also show that these high confidence pairs contain most of the true positives as compared to the expert annotations, a minimal number of false positives that the hybrid approach identifies but were not identified by the experts and omit a minimal number of false negatives, that were identified by the expert annotators but missed by the hybrid approach.

The lexicon produced mixed results with respect to RQ3, which asks about the lexicon's utility in finding missing annotations. The lexicon increased recall, but at a high cost of precision, because the lexicon lacks contextual cues to distinguish when particular action and information type phrases are true positives. In response to RQ4 about lexicon reuse and saturation, we observed that the lexicon reaches a saturation limit of between 42-84% in the domain of privacy policies, which suggests the lexicon will likely never become complete. Alternatively, the lexicon may be used to find annotations for common words and phrases that can be used to further reduce the number of tasks sent to crowd workers and thus the overall framework cost, or can be used to solicit a higher number of workers to complete the reduced number of tasks for the same cost, thus reducing the probability of false negatives.

In response to RQ5, which asks whether crowd workers can reduce these false positives, we observed an improvement in precision as we sent the selected high confidence pairs back to the crowd for acceptance or rejection. Improvements in the front end of the framework (steps 3, 4 and B1 or B2) could further reduce the number of pairs that need to be sent to the workers in step 5, further improving the overall performance.

Based on our earlier work in crowd worker goal annotations [Breaux and Schaub 2014], we estimate our target cost for extracting privacy goals from policies to be $0.92 per statement, which is the cost of two experts to annotate the modality, information actions, information types, sources, targets and purposes. There is an additional $0.18 to have two experts formalize a statement, which aligns with task re-composition. Our results in Section 4 yield a partial specification (excluding the purposes) that costs $1.00 per sentence, which is still under the $1.10 total cost per statement that we obtained with expert annotators, but which excludes the work to identify and link purposes to the goal specifications. One promising observation is that purposes are fairly sparse in the data set and conform to a particular phrase structure that may make them more amenable to automatic detection using phrase structure grammar patterns or machine learning. In addition, a significant portion of the increased cost comes from the validation task (step 5 in the Figure 1). As shown in Table XI, there are still a large number of false positives in the direct and indirect high confidence pairs. To the extent that we can reduce these false positives, we can further reduce the cost of crowd worker extraction.

In our study, we examined privacy policies, which are legal documents intended to be read by various stakeholders, including website users, customers visiting "brick-and-mortar" physical stores, and by regulators. Although we cannot say with certainty, we believe that this approach could perform well on other information-

related policies and laws. In steps 3 and 4 of our framework in Figure 1, we rely on crowd workers to read and interpret a text segment and to identify the actions and information types in the text. With respect to complex laws such as the Privacy Rule of the Health Insurance Portability and Accountability Act (HIPAA), our approach will be limited, because such laws rely on readers to integrate text from across multiple sections to frame the context (e.g., definitions and cross references to exceptions). Therefore, to prepare the inputs for crowd workers, a trained analyst would need to "decontextualize" the legal text by systematically integrating relevant content from related sections of the law. If successful, however, we found that the crowd workers were able to identify both the implicit (e.g.: *provide, access*) and explicit (e.g.: *collect, use, personal information*) types of actions and information types with high recall as compared to the expert annotations. This variation in language use may be more frequent in policies than in laws, in which case, crowd workers may have an easier time identifying relevant noun and verb phrases in laws. For example, the narrower and more consistent use of terminology found in laws may improve the results of step B1 in Figure 1, which consists of the automation achieved by reusing a lexicon of established terminology. The lexicon may reach better saturation across multiple laws than what we observed in policies (see Figure 6), if those laws all adopt similar terms in their regulatory codes.

In summary, we introduced and evaluated a method that combines crowdsourcing and natural language processing (NLP) to extract goals from privacy policies. The results show that crowd workers provide human interpretations that are still beyond the state of the art in NLP, and that for problems with predictable characteristics based on lexical or syntactic features, the NLP provides a cost-effective means to scale the extraction to a larger number of documents. We observed, however, that neither approach can be used by itself: the crowd workers vary in their interpretations and they are prone to miss or overlook information, whereas the NLP techniques studied cannot differentiate among the semantic cues needed to identify relevant phrases without the crowd workers. In the end, we observed that both classes of techniques are complementary and can be used to address each other's weaknesses for improved performance. While we believe this technique could be applied to other domains with similar results, future work is needed to evaluate our approach in such domains.

ACKNOWLEDGMENTS

We thank Hanan Hibshi, Sepideh Ghanavati, Daniel Smullen, Sudarshan Wadkar and the CMU Requirements Engineering Lab for their helpful feedback. This research was funded by the NSF Award CNS-1330596.

REFERENCES

Russell J. Abbott. 1983. Program design by informal English descriptions. *Communications of the ACM* 26, 882-894. DOI: http://dx.doi.org/10.1145/182.358441.

Annie I. Antón. 1997. *Goal Identification and Refinement in the Specification of Information Systems*. Ph.D. Thesis. Georgia Institute of Technology, Georgia, USA.

Annie I. Antón and Julia B. Earp. 2004. A requirements taxonomy for reducing website privacy vulnerabilities. *Requirements Engineering Journal* 9, 3 (August 2004), 169-185. DOI: http://dx.doi.org/10.1007/s00766-003-0183-z.

Mohammad Allahbakhsh, Boualem Benatallah, Aleksandar Ignjatovic, Hamid Reza Motahari-Nezhad, Elisa Bertino, and Schahram Dustdar. 2013. Quality Control in Crowdsourcing Systems: Issues and Directions. *IEEE Internet Computing* 17, 2 (March 2013), 76-81. DOI: http://dx.doi.org/10.1109/MIC.2013.20.

Paul André, Aniket Kittur, and Steven P. Dow. 2014. Crowd synthesis: extracting categories and clusters from complex data. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work and Social Computing*. ACM, New York, NY, USA, 989-998. DOI: http://dx.doi.org/10.1145/2531602.2531653.

Daniel M. Berry, Erik Kamsties, and Michael M. Krieger. 2003. *From contract drafting to software specification: Linguistic sources of ambiguity*. Technical Report. University of Waterloo, Ontario, Canada.

Daniel M. Berry, Ricardo Gacitua, Pete Sawyer, and Sri Fatimah Tjong. 2012. The case for dumb requirements engineering tools. In *Proceedings of the 18th International Conference on Requirements Engineering: Foundation for Software Quality*, Björn Regnell and Daniela Damian (Eds.). Springer-Verlag, Berlin, Heidelberg, 211-217. DOI:10.1007/978-3-642-28714-5_18.

Jaspreet Bhatia and Travis D. Breaux. 2015. Towards an Information Type Lexicon for Privacy Policies. *8th IEEE International Workshop on Requirements Engineering and Law*. IEEE Computer Society, Washington, D.C., 19-24. DOI: 10.1109/RELAW.2015.7330207.

Travis D. Breaux and Florian Schaub. 2014. Scaling requirements extraction to the crowd: experiments on privacy policies. *22nd IEEE International Requirements Engineering Conference*. IEEE Computer Society, Washington, D.C., 163-172. DOI:10.1109/RE.2014.6912258.

Travis D. Breaux, Annie I. Antón, and Jon Doyle. 2008. Semantic parameterization: A process for modeling domain descriptions. *ACM Transactions on Software Engineering Methodology* 18, 2, Article 5 (November 2008), 27 pages. DOI: http://dx.doi.org/10.1145/1416563.1416565.

Travis D. Breaux, Hanan Hibshi, and Ashwini Rao. 2014. Eddy, a formal language for specifying and analyzing data flow specifications for conflicting privacy requirements. *Requirements Engineering Journal* 19, 3 (September 2014), 281-307. DOI: http://dx.doi.org/10.1007/s00766-013-0190-7.

Travis D. Breaux, Daniel Smullen, and Hanan Hibshi. 2015. Detecting Repurposing and Over-collection in Multi-Party Privacy Requirements Specifications. *23rd IEEE International Requirements Engineering Conference*. IEEE Computer Society, Washington, D.C., 166-175. DOI: 10.1109/RE.2015.7320419.

Erik Casagrande, Selamawit Woldeamlak, Wei Lee Woon, H. H. Zeineldin, and Davor Svetinovic. 2014. NLP-KAOS for Systems Goal Elicitation: Smart Metering System Case Study. *IEEE Transactions in Software Engineering* 40, 10, 941-956. DOI:10.1109/TSE.2014.2339811.

Michelene T. H. Chi, Paul J. Feltovich, and Robert Glaser. 1981. Categorization and representation of physics problems by experts and novices. *Cognitive Science* 5, 2, 121–152. DOI: 10.1207/s15516709cog0502_2

Anne Dardenne, Axel van Lamsweerde, and Stephen Fickas. 1993. Goal-directed requirements acquisition. In *Selected Papers of the Sixth International Workshop on Software Specification and Design*, M. Sintzoff, C. Ghezzi, and G.-C. Roman (Eds.). Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, 3-50. DOI:10.1016/0167-6423(93)90021-G

Nguyen Q. V. Hung, Nguyen T. Tam, Lam N. Tran, and Karl Aberer. 2013. An evaluation of aggregation techniques in crowdsourcing. *Web Information Systems Engineering*. Springer Berlin Heidelberg, 1-15. DOI:10.1007/978-3-642-41154-0_1

Silvia Ingolfo, Ivan Jureta, Alberto Siena, Anna Perini, and Angelo Susi. 2014. Nòmos 3: Legal compliance of roles and requirements. *Conceptual Modeling LNCS* 8824: 275-288. DOI: 10.1007/978-3-319-12206-9_22

John S. Justeson and Slava M. Katz, 1995. Technical terminology: some linguistic properties and an algorithm for identification in text. In *Natural Language Engineering* 1, 1, 9-27. DOI:10.1017/S1351324900000048

Erik Kamsties. 2006. Understanding Ambiguity in Requirements Engineering. *Engineering and Managing Software Requirements*. Springer, The Netherlands, 245–266. DOI:10.1007/3-540-28244-0_11

Aniket Kittur, Boris Smus, Susheel Khamkar, and Robert E. Kraut. 2011. CrowdForge: crowdsourcing complex work. In *Proceedings of the 24th Annual ACM Symposium on User interface Software and Technology*. ACM, New York, NY, USA, 43-52. DOI: http://dx.doi.org/10.1145/2047196.2047202

Leonid Kof. 2004. Natural language processing for requirements engineering: Applicability to large requirements documents. In *Proceedings of the 19th International Conference on Automated Software Engineering Workshops,* 91–102.

Soo L. Lim and Anthony Finkelstein. 2012. StakeRare: Using social networks and collaborative filtering for large-scale requirements elicitation. *IEEE Transactions on Software Engineering* 38, 3, 707-735. DOI: http://dx.doi.org/10.1109/TSE.2011.36.

Jialiu Lin, Shahriyar Amini, Jason I. Hong, Norman Sadeh, Janne Lindqvist, and Joy Zhang. 2012. Expectation and purpose: understanding users' mental models of mobile app privacy through crowdsourcing. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. ACM, New York, NY, USA, 501-510. DOI: http://dx.doi.org/10.1145/2370216.2370290

Lin Liu, Tianying Li, and Xiaoxi Kou. 2014. Eliciting relations from natural language requirements documents based on linguistic and statistical analysis. *38th IEEE Annual Computer Software and Applications Conference*, 191-200. DOI: 10.1109/COMPSAC.2014.27

Jeremy C. Maxwell and Annie Antón. 2009. Developing production rule models to aid in acquiring requirements from legal texts. *17th IEEE International Requirements Engineering Conference*. IEEE Computer Society, Washington, D.C., 101-110. DOI: 10.1109/RE.2009.21.

Aleecia M. McDonald and Lorrie F. Cranor. 2008. The Cost of Reading Privacy Policies. *I/S: A Journal of Law and Policy for the Information Society* 4, 3, 540-565.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of International Conference on Language Resources and Evaluation,* 449-454.

Paul N. Otto, Annie I. Anton, and David L. Baumer. 2007. The ChoicePoint Dilemma: How Data Brokers Should Handle the Privacy of Personal Information. *IEEE Security and Privacy* 5, 5, 15-23. DOI: 10.1109/MSP.2007.126

Daniel Popescu, Spencer Rugaber, Nenad Medvidovic and Daniel M. Berry. 2008. Reducing ambiguities in requirements specifications via automatically created object-oriented models. In *Innovations for Requirement Analysis. From Stakeholders' Needs to Formal Designs*, Barbara Paech and Craig Martell (Eds.). Lecture Notes In Computer Science, Vol. 5320. Springer-Verlag, Berlin, Heidelberg 103-124. DOI: http://dx.doi.org/10.1007/978-3-540-89778-1_10.

Elda Paja, Fabiano Dalpiaz, and Paolo Giorgini. 2015. Modeling and reasoning about security requirements in socio-technical systems. *Data Knowledge Engineering* 98, C, 123-143. DOI: http://dx.doi.org/10.1016/j.datak.2015.07.007.

Alexander J. Quinn and Benjamin B. Bederson. 2011. Human computation: a survey and taxonomy of a growing field. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 1403-1412. DOI: http://dx.doi.org/10.1145/1978942.1979148.

J.R. Reidenberg, T.D. Breaux, L.F. Cranor, B. French, A. Grannis, J.T. Graves, F. Liu, A.M. McDonald, T.B. Norton, R. Ramanath, N.C. Russell, N. Sadeh, F. Schaub. 2015. Disagreeable Privacy Policies: Mismatches between Meaning and Users' Understanding. *Berkeley Technology Law Journal* 30, 1, 39-88.

Eleanor Rosch, Carolyn B. Mervisa, Wayne D Gray, David M Johnson, and Penny Boyes-Braem. 1976. Basic objects in natural categories. *Cognitive Psychology* 8, 3, 382–439. DOI: 10.1016/0010-0285(76)90013-X

Marta Sabou, Kalina Bontcheva, and Arno Scharl. 2012. Crowdsourcing research opportunities: lessons from natural language processing. In *Proceedings of the 12th International Conference on Knowledge Management and Knowledge Technologies*. ACM, New York, NY, USA, Article 17, 8 pages. DOI: http://dx.doi.org/10.1145/2362456.2362479.

Patrick Shafto and John D. Coley. Development of categorization and reasoning in the natural world: Novices to experts, naive similarity to ecological knowledge. *Journal of Experimental Psychology: Learning Memory and Cognition* 29, 4, 641-649. DOI: 10.1037/0278-7393.29.4.641.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast-but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Stroudsburg, PA, USA, 254-263.

James W. Tanaka and Marjorie Taylor. 1991. Object categories and expertise: Is the basic level in the eye of the beholder?. *Cognitive Psychology* 23, 3, 457–482.

Vasilis Verroios and Michael S. Bernstein. 2014. Context trees: crowdsourcing global understanding from local views. In *2nd AAAI Conference on Human Computation and Crowdsourcing,* 210-219.

Wesley Willett, Jeffrey Heer, and Maneesh Agrawala. 2012. Strategies for crowdsourcing social data analysis. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 227-236. DOI: http://dx.doi.org/10.1145/2207676.2207709.

Hui Yang, Alistair Willis, Anne De Roeck, and Bashar Nuseibeh. 2010. Automatic detection of nocuous coordination ambiguities in natural language requirements. In *Proceedings of the IEEE/ACM International Conference on Automated Software Engineering*. ACM, New York, NY, USA, 53-62. DOI:http://dx.doi.org/10.1145/1858996.1859007.

Hui Yang, Anne de Roeck, Vincenzo Gervasi, Alistair Willis, and Bashar Nuseibeh. 2011. Analysing anaphoric ambiguity in natural language requirements. *Requirements Engineering Journal* 16, 3, 163-189. DOI:http://dx.doi.org/10.1007/s00766-011-0119-y.

Robert. K. Yin. 2009. Case study research (4th edition). In *Applied Social Research Methods Series*, v.5. Sage Publications.

Jing Zhang, Xindong Wu, and Victor S. Sheng. 2013. A threshold method for imbalanced multiple noisy labeling. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. ACM, New York, NY, USA, 61-65. DOI:http://dx.doi.org/10.1145/2492517.2492640.