Automatic Extraction of Access Control Policies from Natural Language Documents

Masoud Narouei, Hassan Takabi, *Member, IEEE*, and Rodney Nielsen

Abstract—A fundamental management responsibility is securing information systems. Almost all applications that deal with safety, privacy, or defense include some form of access control. There are a plethora of access control models in the information security realm such as role-based access control and attribute-based access control. However, the initial development of access control policies (ACPs) can be very challenging. Most organizations have high-level requirement specifications that include a set of ACPs, which describe allowable operations of the system. It is time consuming and error-prone to manually sift through these documents and extract ACPs. In this paper, we propose a new framework towards extracting ACPs from unrestricted natural language documents using semantic role labeling (SRL). We were able to correctly identify ACP elements with an average F_1 score of 75 percent, which bested the previous work by 15 percent. Furthermore, as SRL tools are often trained on publicly available corpora such as Wall Street Journal, we investigated the idea of improving SRL performance using domain-related knowledge. We utilized domain adaptation and semi-supervised learning techniques and were able to improve the SRL performance by 2 percent using only a small amount of access control data.

Index Terms—Access control policy, policy engineering, semantic role labeling, domain adaptation, semi-supervised learning, natural language processing, transfer learning

1 Introduction

Access control is an essential component of every organization and critical to the security IT systems. Access to these systems and manipulating their data is appropriately controlled according to data classification levels described in access control policies (ACPs). ACPs define permissions to files and folders, user account privileges, firewall permissions, database access rights, encryption, and almost any permission required to access an information system. They also detail management of a number of key issues such as passwords, operating system software controls, higher-risk system access, controlling remote user access, and restricting access. Almost all applications that deal with safety, privacy, or defense include some form of access control. However, defining proper ACPs is challenging, especially for large organizations.

Advanced access control models such as attribute-based access control (ABAC) and role-based access control (RBAC) promise long-term cost savings through reduced management effort. RBAC is the most widely used model for advanced access control in diverse enterprises of all sizes. In RBAC, access permissions are associated with roles instead of users and they represent functions within a given organization. Users can activate a subset of the roles which they are members of and easily acquire all the required permissions for those roles. ABAC is an access control model wherein the access control decisions are made based on a set of attributes, associated with the requester, the environment, and/or the resource itself. An attribute is a property expressed as a name:value pair that can capture identities and access control lists (DAC), security labels, clearances and classifications (MAC) and roles (RBAC). ABAC was proposed as a general model that could overcome the limitations of the dominant access control models (i.e, discretionary-DAC, mandatory-MAC and role-based-RBAC) while unifying their advantages.

Prior to deploying these access control models, an organization needs to identify its initial set of policies. Despite the advantages these models provide, manual development of initial policies can be difficult, expensive, labor-intensive, and error prone. Because of the large number of business processes, users, and permissions in an organization, such a process is a rather difficult task, human intensive, and believed to be slow, and not scalable [1], [2]. Most of the organizations have high-level requirement specifications that are normally expressed in human understandable terms and hence not directly implementable in an access control mechanism. These documents are a rich source of information that could be used in the process of developing initial policies. We refer to these documents as natural language access control policies (NLACPs), which are defined as "statements governing management and access of enterprise objects. NLACPs are human expressions that can be translated to machine-enforceable access control policies" [1]. These documents are unstructured and may be ambiguous and thus hard to convert to formally actionable elements. Therefore, the enterprise policy may be difficult to encode in a machine-enforceable form. In order to properly

• *The authors are with the Department of Computer Sciene and Engineering, University of North Texas, Denton, TX 76203. E-mail: masoudnarouei@my.unt.edu, {takabi, rodney.nielsen}@unt.edu.*

Manuscript received 29 May 2017; revised 30 Nov. 2017; accepted 15 Mar. 2018. Date of publication 23 Mar. 2018; date of current version 13 May 2020. (Corresponding author: Masoud Narouei.)
Digital Object Identifier no. 10.1109/TDSC.2018.2818708

enforce the security policies, these ACPs should be translated to machine-readable policies, which is done manually and is a labor intensive and error prone process [2].

Our goal is to automate this process to reduce manual effort and human error. We propose to develop techniques and tools that will support effective development of trustworthy ACPs through automatically extracting formal ACPs from unrestricted natural language documents. This will allow organizations to use existing natural language texts such as requirements documents for inferring ACPs. We propose to use semantic role labeling (SRL) [3] as a suitable method of identifying policy elements from NLACPs. SRL extracts contextual information and environment conditions, which is beneficial in the proper definition of other access control models such as ABAC.

In the past years, supervised learning algorithms have received significant attention for SRL research, especially because of the availability of manually labeled corpora, such as PropBank [4] and FrameNet [5]. SRL is addressed as a multi-class classification task in which features are generated from an annotated corpus. Several machine learning approaches have been proposed for SRL such as decision trees (DT) [6] and Support Vector Machines (SVMs) [7]. In this paper, we will also compare the performance of the best SRL systems in properly identifying ACP elements. A current limitation of real-world SRL application is its domain dependence. PropBank's text was taken from the *Wall Street Journal* and includes a significant amount of financial news. FrameNet covers a wider variety of text, as its material is taken from the genre-balanced *British National Corpus*, but still suffers the shortcoming of only including predicates from certain predetermined semantic frames. As a consequence, applying current SRL systems on other domains yields lower performance than desired since each domain has its own predicate-argument structure and often these predicates are not found in annotated corpora like PropBank and FrameNet. For example, Pradhan et al. tested SVMs trained on PropBank data on a new test set from the AQUAINT corpus. Switching domains resulted in around a 10 percent drop in precision and recall [8]. To address this issue, we utilize domain adaptation and semi-supervised learning. Specifically, in addition to comparing the performance of various SRL systems, we will also provide a proof of concept of how to adapt these systems to our specific domain in order to improve their performance. This will be done by rebuilding the SRL system's model using domain-related knowledge. Our goal is to allow organizations to use existing, unconstrained natural language texts such as requirements documents for inferring ACPs. Our approach could be used as a standalone top-down approach or as a hybrid approach in combination with bottom-up policy mining approaches.

To the best of our knowledge, there is not much work in the literature that addresses this issue and this is the first report on the effectiveness of applying SRL to a large and diverse set of ACPs.

This paper makes the following main contributions:

- We introduce SRL as a means to identify ACPs in unrestricted NLACPs.
- We compare the performance of different SRL systems in extracting ACPs from unrestricted natural language documents.
- We take advantage of domain adaptation and semi-supervised learning techniques to further improve the performance of SRL for ACP extraction.
- We perform experiments to show the effectiveness of the proposed approach. Our evaluation results show that we can improve SRL performance by using unlabeled domain data in addition to using existing labeled non-domain data.

The rest of the paper is organized as follows: We start with an overview of previous literature and background information in Section 2. In Section 3, we present our proposed ACP extraction framework and its components. The experiments and results are presented in Section 4, followed by discussions and comparison with literature in Section 5. Finally, the conclusion and future work will be presented in Section 6.

2 Background and Related Work

This section describes the state-of-the-art in NLP techniques and their application for ACPs and related areas as well as top-down and hybrid role engineering approaches in the literature. We also provide an overview of SRL, semi-supervised learning, and domain adaptation.

2.1 NLP Techniques for Privacy Policies

Breaux et al. have manually analyzed privacy policies to map natural language policy statements into frame-based and first-order logic representations [9], [10]. They have also analyzed regulatory text and developed natural language heuristics, some expressible as simple regular expressions, which can be used to identify frame-based representations of actions [11] and whether actions on information are permitted, required, or prohibited with various conditions, exceptions, and purposes [12]. Ammar et al. conducted an experiment to use NLP methods and crowdsourced annotations from the "Terms of Service; Didn't Read" project to train a classifier to answer a single question: whether a privacy policy is considered clear (by humans) about a particular set of procedures pertaining to sensitive user data [13]. The ongoing Usable Privacy Policy Project aims to build on recent advances in NLP, privacy preference modeling, crowdsourcing, and formal methods to semi-automatically extract key privacy policy features from natural language website privacy policies [14]. The focus of this project is website privacy policies while our project is focused on ACPs.

2.2 Controlled Natural Language and Access Control

Schwitter defined a controlled natural language (CNL) as "an engineered subset of a natural language whose grammar and vocabulary have been restricted in a systematic way in order to reduce both ambiguity and complexity of full natural language" [15]. While a CNL provides semantic interpretations, it limits policy authors to the defined grammar and requires language-specific tools to stay within the language constraints. The SPARCLE Policy Workbench [16], [17] employs shallow parsing technology to extract

privacy policies based on a pre-defined controlled grammar for forming policies in a structured form. The policies are then translated to a machine-readable form, such as XACML [18]. Inglesant et al. proposed a similar tool, PERMIS, which used a role-based authorization model [19]. However, they reported issues with users not comprehending the predefined building blocks imposed by using a CNL. Recently, Shi and Chadwick [20] presented another tool to write static rules using a CNL. Their tool did not support conditions such as previous actions that must be taken before a user could access data. Our methodology removes all the constraints and process original, unconstrained texts.

2.3 NLP and Access Control

Natural language sources have been analyzed to infer and generate ACPs. Fernandez et al. presented a basic overview of extracting RBAC from use cases [21]. Fontaine proposed an approach based upon goal-based requirements engineering to extract authorization and obligation rules from natural language texts into a policy language [22]. He et al. proposed an approach to generate ACPs from natural language based upon available project documents, database design, and existing rules [23]. Using a series of heuristics, developers manually analyze the documents to find ACPs whereas our approach seeks to automatically extract ACPs. Recent approaches have taken advantage of manually labeled data and predefined patterns to find ACPs. Xiao et al. proposed Text2Policy [24], which uses shallow parsing techniques with finite state transducers to categorize a sentence into one of four possible access control patterns. If such a match can be made, it uses the annotated portions of the sentences to extract the subject, action, and object from the sentence. However, using predefined patterns requires a globally accepted language for describing policies and currently there is no such consensus. In contrast, the semi-supervised learning approach proposed in this paper does not rely on manually defined patterns, but instead learns the patterns using unlabeled data. They also did not take into account the presence of contextual information or environment conditions, which is a challenging task. Slankas et al. proposed access control rule extraction (ACRE) [25], which applies inductive reasoning to find and extract ACRs while Text2Policy applies deductive reasoning based upon existing rules to find and extract ACRs. While these two early works are promising, they suffer from several weaknesses. ACRE uses a supervised learning approach to identify sentences containing ACRs, which requires a labeled dataset similar in structure and content to the document being analyzed. This data is hard to come by. Text2Policy does not require a labeled data set, but it misses ACRs that do not follow one of its four patterns. It is reported that only 34.4 percent of the identified ACR sentences followed one of Text2Policy's patterns [25]. Additionally, Text2Policy's natural language parser requires splitting longer sentences as the parser cannot handle complicated sentence structures. These approaches assume all necessary information for an ACP is contained within the same sentence, and they do not handle resolution issues. Neither one of these approaches take into account the presence of contextual information or environment conditions, which is a very challenging task.

2.4 Semantic Role Labeling

A key part of comprehending natural language is understanding events and their participants: who, what, when, where, etc. Generating this shallow semantic representation of the sentence is called semantic role labeling. SRL consists of detection of semantic arguments associated with a verb (or more generally, a predicate) in a sentence and their classification into specific roles. SRL often labels verb-argument structure using the notation defined by the Propbank [4] project, identifying who did what to whom by assigning roles to constituents of the sentence representing entities related to a specific verb. These semantic arguments are a key to find the answers to questions such as who, when, what, where, why, etc., which are particularly important in extracting ACPs from sentences.

The following sentence exemplifies the annotation of semantic roles:

[Arg_0 John] [$ArgM$-MOD can] [V **assign**] [Arg_1 clerk] [Arg_2 to users from department A]

Here, the roles for the predicate *assign* (assign.01, that is, the roleset of the predicate) are defined in the PropBank Frames scheme as:

- V: *verb*
- $ArgM$-**MOD**: *modal*
- Arg_0: *assigner*
- Arg_1: *thing assigned*
- Arg_2: *assigned to*

Another approach for structuring SRL is FrameNet [5], which is focused on semantic frames. Semantic frames are defined as a schematic representation of situations involving various participants, props, and other conceptual roles [26]. While both Propbank and FrameNet attempt to label semantically related verbs, PropBank is mostly aimed at providing data for training statistical systems [4].

The following sections present the various SRL systems used in the experiments.

2.4.1 SENNA

Semantic/syntactic Extraction using a Neural Network Architecture (SENNA) is a SRL program trained on the PropBank corpus [27]. SENNA is a multilayer neural network architecture that can handle a number of NLP tasks such as POS tagging, chunking, NER, and SRL with both speed and accuracy. It does not rely on the extraction of syntax trees for assigning semantic roles to sentence constituents and instead uses a radically different approach compared to the existing SRL systems: By skipping the time consuming step of generating syntax trees, SENNA's neural network architecture was trained directly on some basic, quickly derivable sentence features. SENNA's output is a sentence annotated with PropBank arguments delivering roles such as subject, argument, negation, location, manner, and others. Depending on the sentence length, the parsing speed of SENNA is between 25 and 390 ms/sentence [28].

2.4.2 SwiRL: The Semantic Role Labeler

SwiRL is a SRL system for English constructed on top of full syntactic analysis of text [29]. The syntactic analysis is performed using the Charniak parser [30]. SwiRL trains one

classifier for each argument label using a rich set of syntactic and semantic features. The classifiers are learned using a one-vs-all AdaBoost algorithm. SwiRL provides a large array of syntactic and semantic analysis. It has strong performance and ranked fifth among the systems that participated at the CoNLL-2005 shared task evaluation [31]. It is fairly robust and can work with both case-sensitive and case-insensitive text.[1] A major advantage of SwiRL compared to other SRL systems is its trainability, which is important when working with domain-specific data.

2.4.3 EasySRL

EasySRL builds logical forms for natural language sentences, by jointly modelling combinatory categorial grammar (CCG) and SRL [32]. CCG is a lexicalized grammar formalism, which associates words with lexical categories [33]. EasySRL uses an efficient A* parsing algorithm, meaning it can be used to process large corpora.

2.4.4 Mate-Tools Semantic Role Labeler

Mate-tools Semantic Role Labeler consists of a three-stage analysis that uses the output of a dependency parser to identify the arguments of the predicates in a sentence [34]. The first stage consists of a set of independent classifiers. They carried out the predicate disambiguation with a set of greedy classifiers, where one classifier is trained for each lemma occurring in the training data. Then using a beam search to identify the arguments of each predicate and to label them yielded a pool of candidate propositions. The second stage consists of a reranker that they applied to the candidates using the local models and proposition features. They combined the score of the greedy classifiers and the reranker in a third stage to select the best candidate proposition. They evaluated their semantic parser on a set of seven languages (Catalan, Spanish, Chinese, Czech, English, German and Japanese) provided by the organizers of the CoNLL-2009 shared task. The proposed system achieved an average labeled semantic F_1 of 80.31, which corresponded to the second best SRL score overall in CoNLL-2009.

2.5 Semi-Supervised Learning

Semi-supervised learning is a class of machine learning tasks and techniques that is concerned with the study of how natural systems such as humans and computers learn in the presence of both labeled and unlabeled data [35]. Learning is usually studied in either a supervised paradigm (e.g., classification) where all of the training data is labeled, or in an unsupervised paradigm (e.g., clustering, outlier detection) where all the data is unlabeled. However, these settings are not optimal in all the cases. There are several cases in which it is not practical to label enough data to employ a supervised paradigm and it would not be effective or appropriate to employ unsupervised learning. ACP extraction is an example where we have a limited amount of labeled domain data but a lot of unlabeled data, and we need to take advantage of both labeled and unlabeled data to have an accurate prediction. Semi-supervised learning is attractive here because it can potentially utilize both labeled and unlabeled data to achieve better performance than supervised learning utilizing only the labeled data [35].

Semi-supervised learning makes use of both labeled and unlabeled data for training typically a small amount of labeled data with a large amount of unlabeled data. It falls between unsupervised learning (without any labeled training data) and supervised learning (with only labeled training data). The goal is to understand how combining labeled and unlabeled data may change the learning behavior, and design algorithms that take advantage of such a combination. The intuition behind using semi-supervised learning is that the acquisition of labeled data for a learning problem often requires a skilled human agent and the cost associated with the labeling process thus may render a fully labeled training set infeasible, whereas acquisition of unlabeled data is relatively inexpensive. From a different perspective, semi-supervised learning may achieve the same level of performance as supervised learning but with fewer labeled instances. This reduces the annotation effort, which leads to reduced cost.

2.6 Domain Adaptation

The task of domain adaptation is developing learning algorithms that can be easily ported from one domain to another-say, for example, from *newswire* to software requirements documents. This scenario arises when we aim to adapt a classifier learned from a source data distribution to a different target distribution. This issue is particularly interesting in NLP because we are often in the situation that we have a large collection of labeled data in one source domain (say, *Wall Street Journal* news articles) but truly desire a model that performs well in a second target domain such as requirement documents.

The problem of domain-dependence in SRL systems has been previously studied by He and Gildea through bootstrapping unlabeled data in new domains [36]. They explored the possibility of a weakly supervised approach by using self-training and co-training, making use of both labeled and unlabeled data and comparing two machine learning techniques, decision lists, and maximum entropy. They found that while the decision list system is designed to take advantage of distinct features in order to enable the "two views" of the co-training algorithm to bootstrap one another, it did not show signicant improvement from adding unlabeled data. They found that the higher overall performance of the maximum entropy system outweighed any benefits from the decision list approach [36].

3 The Proposed ACP Extraction Framework

In order to construct a formal model for an NLACP, we must extract the necessary elements of ACPs from the NLACPs. The ACPs describe who has access to what resource in what way. By processing these documents, our methodology will extract ACP elements. An overall view of the proposed methodology is shown in Fig. 1. In the following sections, we describe each of these steps in detail.

3.1 Lexical Parser

In order for the input text to be ready for evaluation, the first step is to identify all sentences and separate them by a carriage return, so that each sentence will be on a separate line. We read each NLACP entirely and perform sentence

1. http://www.surdeanu.info/mihai/swirl/

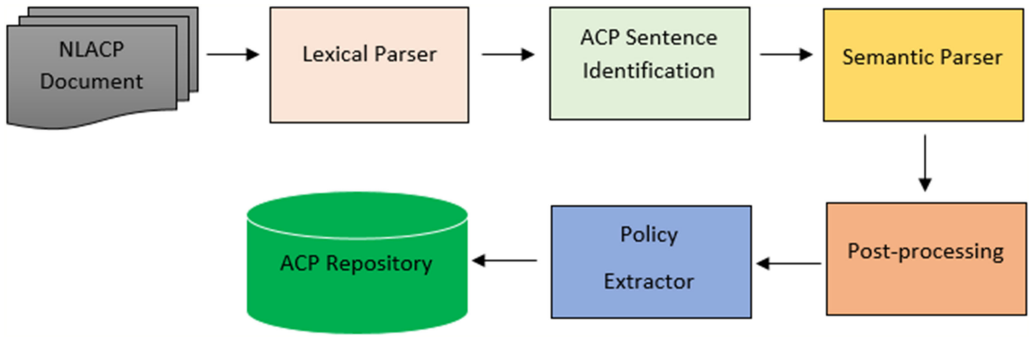

Fig. 1. Overview of the proposed framework.

segmentation and tokenization. Sentence segmentation identifies the boundaries of sentences whereas Tokenization detects individual words, punctuation, and other items from the text. For this purpose, we use the CoreNLP tool kit [37].

3.2 Coreference Resolution

Coreference resolution (sometimes written co-reference) determines whether or not two expressions in a document refer to the same entity or event. The goal is to identify all expressions that refer to the same entity in a text. For example, consider the following sentence where HCP stands for healthcare professional:

The *HCP* opens the message to which *he* or *she* wishes to reply.

Here, "HCP", "he", and "she" all refer to the same entity. The goal of coreference resolution is to decide which mentions or references, whether pronominal or full noun phrases, are referring to the same real world referents. Because each sentence will be evaluated separately, having a clear idea of each pronoun is a key point in identifying the correct ACP elements. We adopt the approach proposed in [38], which is a fast and robust algorithm for this purpose.

3.3 ACP Sentence Identification

Often NLACPs contain content that describes functional requirements that are not necessarily related to ACPs. Attempting to extract ACPs from the whole document is an error-prone and tedious process since many of the sentences do not contain ACPs. Therefore, we need to determine which sentences have ACP content and then perform further analysis in order to extract ACP elements on those sentences. Previous literature [25] proposed a k-nearest neighbors (k-NN) classifier to identify sentences containing ACPs. K-NN is an instance-based classifier that attempts to locate the k-nearest neighbors of an instance in an instance space and labeling that instance with the same class as that of most neighbors. Since our focus is mainly on correctly identifying ACP elements and using those ACPs to create various access control models such as attribute based and role based, we use the same sentences identified in previous work as ACP and make no further contribution in this paper for identifying ACP sentences (sentences that describe ACP).

3.4 Semantic Parser

An issue prior to developing an ACP for any access control model is the information that needs to be encoded is typically buried within existing natural language artifacts, hence difficult to interpret. For example, consider the following ACP sentence for iTrust [39], "System displays only the applicable input entries to the UAP." This ACP sentence is not amenable for automated verification, requiring manual effort in extracting the necessary elements (e.g., ACP subject, object, and action) from this sentence. To address this issue, some researchers have proposed approaches for automatically generating machine-enforceable ACPs from natural language software documents in different formats such as eXtensible access control markup language (XACML) [24]. *ACRE* [25] used an iterative algorithm to discover patterns that represent ACP rules in sentences. They seeded this algorithm with frequently occurring nouns matching a subject-action-resource pattern throughout a document. The algorithm then searched for additional combinations of those nouns to discover additional patterns. The instances found were assumed to represent ACPs and the elements of the ACP were then extracted.

In this work, we proposed to use SRL to automatically identify predicate-argument structure in ACP sentences. SRL is very important in making sense of the meaning of a sentence. Such semantic representation is at a higher-level of abstraction than a syntax tree. For instance, the sentence "A professor can review the same project at most one time" has a different syntactic form but the same semantic roles as "The same project can be reviewed by a professor at most one time". In general, given a sentence, the task of SRL consists of analyzing the propositions expressed by all the predicates in the sentence, and for each, determining which constituents in that sentence fill which semantic roles. Here, we use the following notation to describe ACPs:

$$\{A; B; C\}. \tag{1}$$

Where A stands for Arg_0, B stands for *predicate* and C stands for Arg_1. Arg_0 is the PropBank role that usually denotes agent or experiencer for the predicate and Arg_1 denotes the theme (what the predicate affects).

3.5 Post-Processing

ACPs usually do not conform to a predefined template unless there is a controlled grammar being used. After generating predicate-argument structures using the SRL tool, additional processing of the output is required. This is due to the fact that the NLACPs are typically stated by analysts using their own language and grammar (e.g., some sentences contain more than one ACP). In order to increase the specificity of the extracted ACPs, additional processing is required. Consider the following sentence for example:

Customer Service Reps, Pharmacists, and Billing Reps can collect and use customer name and date of birth to help confirm identity.

There are 15 different ACPs associated with this sentence:

- customer service rep; collect; customer name
- customer service rep; collect; customer date of birth
- customer service rep; use; customer name
- customer service rep; use; customer date of birth
- pharmacist; collect; customer name
- pharmacist; collect; customer date of birth
- pharmacist; use; customer name
- pharmacist; use; customer date of birth
- billing rep; collect; customer name
- billing rep; collect; customer date of birth
- billing rep; use; customer name
- billing rep; use; customer date of birth
- customer service rep; confirm; identity
- pharmacist; confirm; identity
- billing rep; confirm; identity

Now consider the following list of the extracted arguments for the predicate *Collect* using an SRL tool:

*[**Arg$_0$** Customer Service Reps, Pharmacists, and Billing Reps] [**ArgM-MOD** can] [v collect] and use [**Arg$_1$** customer name and date of birth] [**ArgM-PNC** to help confirm identity].*

As a comparison between the identified semantic arguments and the actual ACPs shows, SRL's output can be interpreted as an abstract form for ACPs, so we need to expand this abstract form to generate all of the related ACPs. This expansion could be in the form of extracting all named entities and other standalone nouns in the Arg_0 as the ACP subjects and also extracting independent entities from Arg_1 as the ACP objects. For example, in this case, NER identifies *Customer Service Reps* and *Pharmacists* as organizations. After extracting entities, we list all of their combinations for each predicate. For example, the above verb-argument listing can be expanded as the following rules:

- customer service rep; collect; customer name
- customer service rep; collect; customer date of birth
- pharmacist; collect; customer name
- pharmacist; collect; customer date of birth
- billing rep; collect; customer name
- billing rep; collect; customer date of birth

3.6 Policy Extractor

The policy extractor utilizes the ACP components extracted by the SRL tool to define new policies for different access control models. The ACPs are in the form of subject, object, action, where many of the extracted subjects correspond with the job functions within the organization (e.g., doctor, pharmacist, nurse, healthcare professional, etc.), which typically represent roles. Hence, the extracted information can be used to define roles, which can be used to build an RBAC model. A naïve approach would be to just look at the ACPs, find the ones with the same subject, group them together in one role, and use all the ACPs with that subject to build the corresponding role permission assignment relationships. The object and operation elements of the ACPs are used to define permissions in RBAC, which should be assigned to the roles corresponding to the associated ACP subject.

The SRL tools also provide other information such as the non-numbered arguments of $ArgM\text{-}TMP$ and $ArgM\text{-}LOC$, where $ArgM\text{-}TMP$ refers to the time associated with the proposition and $ArgM\text{-}LOC$ refers to it's location. As an example, consider the following sentence:

[Arg_0 The system] [V **retrieves**] [Arg_1 the student information] [$ArgM\text{-}LOC$ in the registration system].

This additional information is often associated with environment attributes and alongside other arguments can be used to build an ABAC model. In conclusion, depending on the required access control model, policy authors can leverage the extracted information to define new policies.

4 Experiments

The evaluation consists of three independent experiments. In the first experiment, we compare the performance of the four SRL systems, described in Section 2.4, in correctly identifying ACP elements. In the second experiment, we analyze ACP sentences using the best performing system from Section 4.3. Finally, the third experiment provides a proof of concept for how to improve the performance of SRL system using domain adaptation and semi-supervised learning techniques. We present and answer the following research questions in this section.

- *RQ1: How effectively can the subject, action, and resource elements of ACPs be extracted from ACP sentences using SRL?*
- *RQ2: Which SRL tool performs better on extracting ACP elements?*
- *RQ3: Can we improve the performance of SRL in the ACP domain?*

4.1 Datasets

We perform our experiments on four dataset(s) that have been used frequently in the literature. These datasets include documents from different domains, specifically, the healthcare, education, and conference management domains. For the healthcare domain, we use an ACP dataset created by Xiao et al. [24] extracted from iTrust [39], an open source healthcare application that includes various features such as maintaining medical history of patients, identifying primary caregivers, storing communications with doctors, and sharing satisfaction results. For the education domain, we employ use cases from the IBM Course Registration System used by Slankas et al. in prior research [25]. For the conference management domain, we use requirements documents from CyberChair [40],[2] which has been used by hundreds of different conferences and workshops. We also use a combined document of 114 ACP sentences collected from 18 sources (published papers, public web sites, etc.) [24]. The total number of ACPs, sentences, and also SRL arguments for each dataset is shown in Table 1. More details on how this labeling was done can be found in [25].

4.2 Evaluation Criteria

Given an ACP sentence, we want to know how effectively the semantic roles of each predicate are extracted. For this

2. http://www.borbala.com/cyberchair/ for more clarification

TABLE 1
Study Document Set Statistics

Document	Domain	Number of ACP Sentences	Number of ACPs	Number of SRL Arguments
iTrust for Text2Policy	Healthcare	418	1,070	1,559
IBM Course Management	Education	169	375	912
CyberChair	Conference Mgmt	139	386	696
Collected ACP Documents	Multiple	114	258	650
Total		840	2,089	3,817

purpose, the results are evaluated with respect to recall, precision, and the F_1 score of the predicate arguments. Recall (R) is the proportion of all gold-standard arguments that are found or predicted by a system, whereas, Precision (P) is the proportion of arguments predicted by a system that are correct. To compute these values, we categorize the classifier's predictions into four categories: True positives (TP) are correct predictions of semantic arguments, True negatives (TN) are predictions where the classifier correctly predicted that a constituent was not a semantic argument of the predicate in question, False positives (FP) are cases where the classifier mistakenly identified a constituent as an ACP argument when it was not, and False negatives (FN) occur when the classifier fails to correctly predict an actual ACP argument. Using these values, Precision is calculated according to $P = \frac{TP}{TP+FP}$ and recall according to $R = \frac{TP}{TP+FN}$. Finally, the F_1 score is the geometric mean of precision and recall, giving an equal weight to recall and precision. F_1 is computed by the following equation:

$$F_1 = 2 \times \frac{P \times R}{P + R}. \tag{2}$$

4.3 Access Control Policy (ACP) Extraction

RQ1: How effectively can the subject, action, and resource elements of ACPs be extracted from ACP sentences using SRL?

4.3.1 Experiment Setup

In order to evaluate the effectiveness of our proposed approach, we use the dataset(s) that were manually labeled by Slankas et al. [25]. They were able to find a total of 1,070 ACPs in the iTrust dataset, 375 ACPs in the IBM Course Registration System dataset, 386 ACPs in the CyberChair dataset, and 258 ACPs in the Collected ACP documents. More details on how the labeling was done can be found in [25]. In this experiment, we apply SENNA on all five datasets. The evaluation results as well as comparison with the most recent system (ACRE) are presented in Table 2.

TABLE 2
Comparison of ACP Extraction Between ACRE and the Proposed System (IBM: IBM Course Registration, CC: CyberChair and CAD: Collected ACP Documents)

	ACRE			SENNA		
Dataset	Precision	Recall	F_1	Precision	Recall	F_1
iTrust	80%	75%	77%	75%	88%	80%
IBM	81%	62%	70%	54%	87%	58%
CC	75%	30%	43%	46%	84%	59%
CAD	68%	18%	29%	79%	86%	82%
Avg	76%	46%	55%	64%	86%	70%

4.3.2 Experimental Results

As the results show, our approach based on SRL performs very well and outperforms the ACRE approach in most cases. The algorithm used in ACRE requires repetition in sentence structure as well as subjects and resources throughout the document to perform well. This algorithm performed best on iTrust because it contained repetitions throughout the document, but performed poorly on the Collected ACP document because there is not enough repetition in that document for finding an initial set of known subjects and resources and expanding the patterns. However, SRL does not require repetition as every sentence will be considered separately, independent of the other sentences. As long as there are role sets defined for that predicate, SRL can find most of the arguments. This is why the results of SRL are more stable throughout all documents regardless of their structure. In terms of precision, however, our approach does not perform very well. One issue with using SRL is that it extracts all arguments for all of the predicates in a sentence. Sometimes only a portion of these verbs such as "set", "add", etc., describe ACPs. Consider the following example:

Only the manager [v **is** *] [v* **allowed** *] to [v* **add** *] a new resident to the system and to [v* **start** *] or [v* **update** *] the care plan of a resident.*

Here, only three of the verbs, namely "add", "start", and "update" address ACPs. In the experiments, we eliminate "To Be" and "Modal" verbs because usually they are part of other verbs such as *can assign* and do not express ACPs on their own. There are also other verbs such as click, include, etc., that do not express ACPs and hence increase the false positive rate. In the future, we plan to create a dictionary of the verbs that are associated with ACPs and will only consider those verbs which will improve the results significantly. Another issue with our approach is that sometimes the SRL tool is unable to correctly identify all predicates and their arguments. This is due to the complex structure of some sentences. This issue, however, can be resolved by retraining the SRL tool using domain adaptation and semi-supervised learning techniques (more on this in next section). Although our approach does not perform very well in terms of precision, if we consider the F_1 scores, we can see that our approach outperforms ACRE and for some dataset (s) the difference is very substantial (82 percent compared to 29 percent for the collected ACP documents). Only for the IBM Course Management dataset, SRL is outperformed by ACRE and it is because the precision is very low, which leads to a lower F_1 score. In addition to offering better recall and F_1 score, another advantage of our approach over

TABLE 3
Comparison of F_1 Scores Using Different SRL Systems (CAD = Collected ACP Documents, CC = CyberChair and IBM = IBM Course Registration)

System	CAD	CC	IBM	iTrust	Overall
SENNA	82.32	59.55	58.02	80.47	70.09
SwiRL	70.34	46.32	36.87	43.95	49.37
EasySRL	53.66	34.83	37.43	57.18	45.78
Mate-tools	46.11	51.87	60.14	59.35	54.37

ACRE is that it does not require any labeled data set whereas ACRE uses a supervised learning approach and requires a labeled dataset similar in structure and content to the document being analyzed to set up the classifiers. One technical challenge concerning the use of SRL is that sometimes our tool is unable to find the arguments in some sentences. The reason is that SRL tools are often trained on corpora from a very different domain, such as the *Wall Street Journal*. This means that the predicate-argument frames are often not well suited for processing information such as access control requirements documents. In the next section, we address this issue by adapting the SRL tool to the ACP domain to improve its predicate-argument coverage.

4.4 Comparing SRL Systems

RQ2: Which SRL tool performs better on extracting ACP elements?

4.4.1 Experiment Setup

In order to compare the performance of different SRL systems, we manually labelled all arguments in all predicates that were previously identified by Slankas [25] and consider it as the gold standard. We convert this labeling to CoNLL-2005 format, alongside necessary corrections such as fixing argument boundaries. Then we feed this data without gold labels to our four SRL systems and predict new labels. SENNA and SwiRL return the output in the CoNLL-2005 format; however, the EasySRL and Mate-tools do not. Hence, we transform their output to CoNLL-2005 format with necessary adjustments. Then we compare the results using the official CoNLL-2005 shared task scoring software. Results in terms of F_1 are presented in Table 3.

4.4.2 Experimental Results

It is worth emphasizing that our goal is not to evaluate SRL systems in the general domain, but rather to focus on their performance on ACP-relevant sentences. The reason for low performance is mainly because of the strict scoring of the official CoNLL-2005 software. The software requires the phrase boundaries of arguments to match exactly, however, since some of the SRL systems only identify parts of an argument, they receive no credit. This is because sometimes they miss word(s) that should have been included as part of the argument, or include word(s) that should not have been part of the argument (for example, sometimes a preposition such as "in" should be included in the argument and sometimes it should not). An issue with transforming the output of Mate-tools is that sometimes different arguments of the same predicate share a word and it is not possible to add all

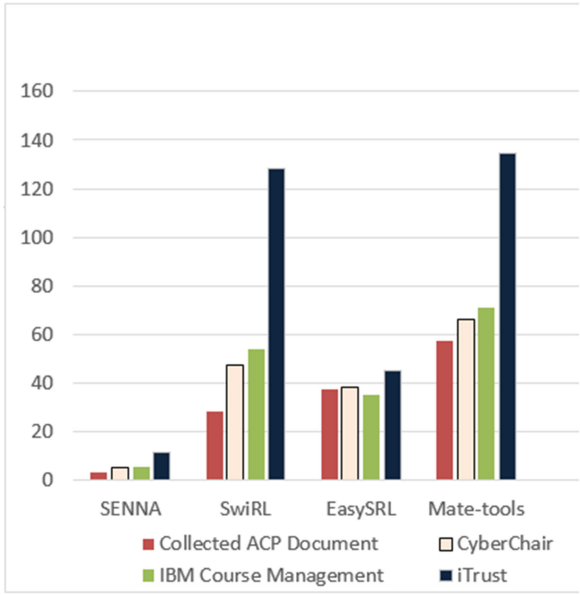

Fig. 2. Comparison of different SRL systems in terms of processing time.

of those arguments to the output given the CoNLL-2005 format, as the scoring tool produces errors; hence, we exclude some words from specific arguments. An issue with EasySRL is that it gives the lemma of each predicate instead of the actual predicate. This causes some issues when there are more than one ACP in a sentence and those ACPs have predicates that have the same lemma (e.g., sends and send in: "The PCC sends the papers and review forms to the reviewers, who must fill in the review forms and send these to the PCC"). Results are not the only consideration in choosing an SRL tool, as processing time can be significant for large documents. Hence, we compare the processing time of these systems in Fig. 2. The time is the total time of processing the whole document plus that of loading required models. It is obvious that SENNA clearly outperforms the other SRL systems. This can be attributed to it's simple architecture, which does not rely on the output of existing NLP systems and also since it was written completely in the C language.

As we mentioned in previous sections, the current SRL tools use a pre-trained model that was generated by learning from out-of-domain data. When we apply these tools on our access control domain, they will not perform as well because of the differences in the grammatical style and word usage. We believe that by retraining the SRL tool's model using domain specific data, we will be able to improve results. As Table 3 and Fig. 2 show, SENNA would appear to be the best choice, as it combines high processing speed with high SRL accuracy. However, after studying the structure of SENNA and contacting the authors, we concluded that SENNA is not easily trainable and would require a time-intensive detailed examination of the source code. Instead, in the following experiment, we take advantage of the user-friendly nature of SwiRL and the ability to easily retrain its model. We chose SwiRL as a proof of concept to show that by using enough domain-specific knowledge, we can significantly improve the performance of the SRL system. This methodology can be applied to any SRL system in order to improve its results.

TABLE 4
Comparison of Domain Adaptation and Semi-Supervised Learning Results

	Precision			Recall			F_1		
Dataset	baseline	DA	SSL	baseline	DA	SSL	baseline	DA	SSL
iTrust	74.87%	80.11%	81.54%	32.43%	33.79%	36.05%	45.25%	47.53%	50%
IBM	60.50%	66.95%	64.41%	23.08%	25.32%	24.36%	33.41%	36.74%	35.35%
CC	79.63%	81.13%	85.71%	22.28%	22.28%	21.76%	34.82%	34.96%	34.71%
CAD	88.49%	91.24%	91.91%	73.65%	74.85%	74.85%	80.39%	82.24%	82.51%

Baseline: initial results obtained from applying SRL tool on testing set, DA: results after applying domain adaptation, SSL: results after applying semi-supervised learning, IBM: IBM Course Registration, CC: CyberChair and CAD: Collected ACP Documents.

4.5 Domain Adaptation and Semi-Supervised Learning

RQ3: Can we improve the performance of SRL in the ACP domain?

4.5.1 Experiment Setup

As discussed in previous sections, applying current SRL systems on other domains yields lower performance than desired since each domain has its own predicate-argument structure and often these predicates are not found in annotated corpora such as PropBank and FrameNet. In this section we use SwiRL to provide a proof of concept for how to eliminate this limitation using a small amount of domain-specific data. SwiRL was originally trained using the PropBank corpus and does not generalize well to the predicate-argument structure in the ACP domain. These differences will yield lower performance when applying SwiRL on ACP datasets. To address this issue, we rebuild SwiRL's model using domain adaptation. For this purpose, we first manually label a small amount of ACP data, add it to SwiRL's initial training set, and then retrain the model. For the semi-supervised learning experiment later in this paper, instead of using manually labeled data, we predict the labels using the original SwiRL model. Then we add this newly labeled data to the initial training set and retrain the model. These techniques and their results are discussed in more detail in the next section. We perform three different experiments with the goal of improving the performance of ACP extraction. We divide each dataset into 70 percent training data and 30 percent test data. For the baseline, we apply the SRL, trained on PropBank, to each test set. The evaluation results are presented in Table 4. As the results show, SwiRL performed reasonably well in terms of precision on most of the datasets.

4.5.2 Using a Simple form of Domain Adaptation to Improve the Performance of ACP Extraction

As a current issue with semantic role labelers, they are not consistent with specific target domains like ACP domain. This is due to the fact that they were trained on publicly available corpora such as PropBank, which was taken from the *Wall Street Journal*. This means that the predicate-argument frames are usually specific to that domain, in this case, largely financial articles. Many real world applications are in need of applying SRL and they have their own datasets expressed in their specific domain. To address this issue, we aim to transfer knowledge from our domain-specific datasets and create a SRL adapted to requirements documents and ACP-specific language in order to achieve better performance on our specific domain. To the best of our knowledge, this is the first report on retraining a semantic role labeler for ACP extraction. As indicated previously, after manually labeling datasets, we divide each dataset into 70 percent training data and 30 percent test data. For this experiment, we add the combined 70 percent training data to the semantic role labeler's original training corpus and retrain it. The evaluation results after applying the newly trained model on the test sets are presented in Table 4 (DA columns). As Table 4 shows, retraining the semantic role labeler yields about a 2 percent increase in F_1 for most of the datasets compared to baseline. Although this was expected, the reason for such a low increase is that the original SwiRL training corpus has over 1 million training examples and the training data that we added to that corpus is roughly 10 thousand examples, about 1 percent of the total size. The original corpus dominates the training phase, and hence the new model is heavily biased by that data. We will investigate this issue in the future by finding a balance between the size of the source (initial) domain and the target domain.

4.5.3 Using Semi-Supervised Learning to Improve the Performance of ACP Extraction

As we mentioned earlier, there is very little labeled data in the ACP domain and previous works are heavily based on using the labeled data. Manually labeling such data is labor-intensive, expensive, and time consuming. In the previous experiments, we took advantage of manually labeled domain-specific data and achieved higher performance, but as the manual labeling is expensive, we decided to propose another solution that has competitive performance without the additional labeling. For this task, we propose to use semi-supervised learning techniques to take advantage of unlabeled ACPs and functional requirements. To do this, we utilize the SwiRL model trained on out-of-domain data to label the unlabeled in-domain training data. With that being said, we ignore all the labels that we have for our training data from human annotation and instead use SwiRL to predict the labels for our training data. We add the automatically labeled data to the semantic role labeler's training corpus. Finally, we retrain SwiRL using this new data and build a new model. Applying the newly trained SwiRL on our test data yields the results presented in Table 4 (SSL columns), which on average are just as good as the results obtained from manually labeling that same data.

4.5.4 Discussion and Comparison with the State-of-the-Art

The trends for recall, precision and F_1 for each dataset are drawn in Fig. 3. The increase suggests improvement in all

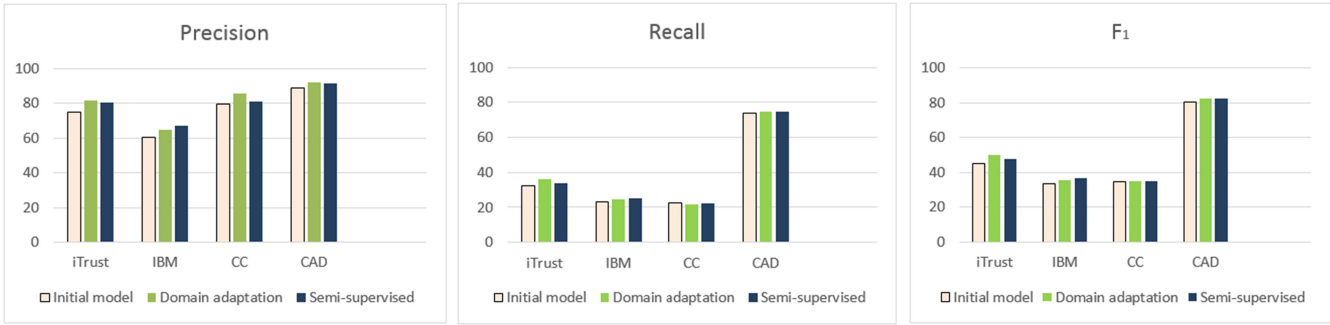

Fig. 3. Precision, recall and F_1 trend analysis.

aspects based on the conducted experiments. Although it is reasonable that using human labeled domain-specific data yields a higher performance than semi-supervised learning in some cases, the difference is slight. The fact that manually labeling dataset(s) is expensive and time-consuming makes the semi-supervised learning results even more appealing. Furthermore, by adding more unlabeled data, we should be able to improve over the performance using gold-standard labeled data.

To further evaluate the performance of our system, we compare the model that was trained using semi-supervised learning with the most recent work, the ACRE system proposed by Slankas [25]. ACRE uses a supervised learning approach and requires a labeled dataset similar in structure and content to the document being analyzed to train the classifiers. The comparison results are shown in Table 5. This experiment is performed at ACP level where we are more concerned with the whole ACP rather than individual arguments. Our proposed system mostly outperforms ACRE in terms of recall and F_1; However, precision seems low. The reason for a lower precision is the fact that the SRL tool produces arguments for all predicates in the sentence, thereby resulting in many false positives. One direction of future work would be to create a list of ACP predicates and then try to filter out SRL results based on the list. Another direction would be to train a classifier to determine which predicates/propositions are relevant. An issue with ACRE is that it requires manually labeled datasets, which are expensive and time-consuming to create. Our goal is to reduce the burden of labeling datasets by using the unlabeled data directly. After getting ACP sentences, another type of labeling (identifying subject, object, action, etc.) is required for extracting policies. Our proposed methodology does not require this labeling as we utilize semi-supervised learning to predict those labels, whereas previous work tried to manually identify the subject, object, and action elements. ACRE performed best on iTrust because it contained repetitions throughout the document but performed poorly on the Collected ACP Document, because there are not enough repetitions in that document for finding the initial set of subject and resource seed patterns, before expanding the patterns using their proposed bootstrapping method. However, since the SRL tool has already been trained using a very large model, it considers each sentence separately, independent of the other sentences. As long as there are rolesets defined for a specific predicate in the initial training set, SRL can identify that predicate's arguments. An instance of this roleset was described in Section 2.4. SwiRL trains on the *Wall Street Journal* data annotated per the PropBank frames/roll sets. PropBank has numerous rolesets defined for each predicate. We effectively added more related rolesets by retraining SwiRL using ACP data.

TABLE 5
Comparison of ACRE and the Proposed System Using Semi-Supervised Learning in Correctly Identifying ACP Elements

	ACRE			SwiRL		
Dataset	Precision	Recall	F_1	Precision	Recall	F_1
iTrust	80%	75%	77%	75%	72%	73%
IBM	81%	62%	70%	69%	75%	72%
CC	75%	30%	43%	63%	54%	56%
CAD	68%	18%	29%	73%	71%	71%

5 Discussion

SRL systems are usually dependent on large quantities of annotated training data. There is lack of annotated data for access control domain and it is both difficult and highly expensive to produce such data. Hence, being able to generalize SRL systems to access control domain could offer a huge benefit. However, generalization of these systems to a new domain different than the domain used to train the system, is a challenging task. A major impediment to the widespread application of SRL across different domains is the degradation of performance, when a supervised SRL system is faced with unseen data. A particular difficulty is that a predicate in the target domain may exhibit a behavior not contemplated in the dictionary of frames at training time. It also remains unclear to what degree generalization is constrained by other components such as tagging and syntactic parsing. Furthermore, the relation of semantic roles to other semantic knowledge (e.g., WordNet, named entities) has barely been addressed in the design of current SRL models. A deeper understanding of these type of questions could help in developing methods that yield improved generalization, and that are less dependent on large quantities of annotated training data.

Our system was able to label semantic roles automatically with fairly high F_1, indicating promise for applications in various security-related natural language tasks. Although SwiRL's current model is fairly accurate on the data covered by the training set, the different nature of the ACP target domain led us to introduce new knowledge sources in order to extend SwiRL's feature set and design a more complex

learning model which in turn resulted in better performance on the target domain. In light of this, there is a question of why adding unlabeled data and retraining the model improves the performance. To answer this question, we need to consider that our classifier learned how to find arguments for each predicate when it was trained using the PropBank corpus, but it is not able to generalize its model in new domains because it simply has not seen instances from the target domain to build a great feature space. What we are looking for is a classifier that generalizes to unseen test documents. The assumption is that when the labels are not available, perhaps they can be guessed. When adding unseen data to the classifier's training data, if we can outperform the class priors, we can find a better classifier using this data. Another benefit of adding unlabeled data is that the classifier finds a new range of feature values due to new data and so will create a better or more general model, thereby improving the generalizability of the model to the new domain.

Our trained models showed a noticeable improvement over our baseline (about 2 percent by domain adaptation and 3 percent by semi-supervised learning). However, this could be increased considering that we have not been able to fully exploit the potential of our models. Still, by only adding a small fraction of in-domain data (additional in-domain data was only about 1 percent of the size of the initial training dataset, thereby increasing the training set size by a mere 1 percent), our system's performance is in line with a state-of-the-art system.

Many aspects of our system are still quite preliminary in identifying ACPs. For example, our system currently assumes knowledge of the correct frame type for the target predicate to determine the semantic roles of its arguments. However, some predicates have several meanings depending on the context being used. Some predicates are also inherently negative (e.g., "prohibit") which is not captured by our current system. Our system is able to identify some negative modal verbs such as "shall not" in "An administrator shall not be permitted to read or write to medical elements of a patient record.", however, it is unable to capture other forms of negation such as "No doctors can view patient records." A more generalized system would thus require a module for frame disambiguation and negative sentence identification.

Much remains to be done to apply the system described here to the interpretation of general policy documents. Our system does not currently identify conditions in ACP sentences, which is critical to the meaning of some ACPs. For example, in "If the invoices total amount exceeds one million, then two different supervisors must authorize the invoice," our system considered "If the invoices total amount exceeds one million" as an adverb for predicate "authorize", which, if not taken into account as a condition, results in an invalid policy. The system also missed on some important arguments in more complex ACP sentences. For example, in "An email alert is sent out to the iTrust user in the event of a changed password, status change in laboratory procedure, comprehensive report requested and generated," the system missed on considering "in the event of a changed password, status change in laboratory procedure, comprehensive report requested and generated". as an argument for predicate "send". This resulted in extracting a policy that is not usable. A more comprehensive training set is thus a necessity. The system also showed inconsistencies in identifying temporal and location arguments. One technique would be the combination of SRL training data with named-entity systems for recognizing entities (e.g., times, dates, and locations), the effort that has gone into recognizing these items should complement the training data.

Generalization to predicates for which no annotated data are available may be possible by incorporating more resources (such as WordNet, FrameNet) or automatic clustering of predicates. Automatically learning generalizations about the semantics and syntactic behavior of predicates is an exciting problem for the years to come. We believe it is critical for the future of SRL that research broadens to include wider investigation of unsupervised methods.

6 Conclusion and Future Work

This paper explores using semantic role labeling as a technique for correctly identifying access control policy elements. We compared four high performance SRL systems. Using the best performing SRL system allowed us to identify ACP elements with an average F_1 score of 75 percent, which bested the previous work by 15 percent. Moreover, as current semantic role labelers are domain dependent due to being trained on specific corpora such as PropBank, applying them to other domains yields lower performances than desired. To address this issue, we proposed using domain adaptation and semi-supervised learning techniques. We were able to increase the performance in terms of recall and F_1 for most of the datasets. It is worth emphasizing that the additional in-domain data only increased the training dataset size by 1 percent. We believe that by adding enough unlabeled data from the ACP requirements domain, we can specifically adapt the tool to perform well on the new domains and outperform the state-of-the-art systems. In the future, we aim to use other domain adaptation and semi-supervised learning techniques to further increase the performance of the SRL tool. We also plan to automate the process of migrating to new access control models such as attribute-based access control using the extracted ACP elements.

Acknowledgments

The authors would like to acknowledge John Slankas for providing the datasets and Mihai Surdeanu for his useful comments on using the Swirl semantic role labeler. We would also like to thank anonymous reviewers for their valuable comments.

References

[1] V. C. Hu, D. Ferraiolo, R. Kuhn, A. R. Friedman, A. J. Lang, M. M. Cogdell, A. Schnitzer, K. Sandlin, R. Miller, K. Scarfone, "Guide to attribute based access control (ABAC) definition and considerations (draft)," *NIST Special Publication*, vol. 800, p. 162, 2014.

[2] M. Beckerle and L. A. Martucci, "Formal definitions for usable access control rule sets from goals to metrics," in *Proc. 9th Symp. Usable Privacy Secur.*, 2013, pp. 2:1–2:11.

[3] D. Gildea and D. Jurafsky, "Automatic labeling of semantic roles," *Comput. Linguistics*, vol. 28, no. 3, pp. 245–288, 2002.

[4] M. Palmer, D. Gildea, and P. Kingsbury, "The proposition bank: An annotated corpus of semantic roles," *Comput. Linguistics*, vol. 31, no. 1, pp. 71–106, 2005.

[5] C. F. Baker, C. J. Fillmore, and J. B. Lowe, "The Berkeley FrameNet project," in *Proc. 36th Annu. Meet. Assoc. Comput. Linguistics 17th Int. Conf. Comput. Linguistics*, 1998, pp. 86–90.

[6] M. Surdeanu, S. Harabagiu, J. Williams, and P. Aarseth, "Using predicate-argument structures for information extraction," in *Proc. 41st Annu. Meet. Assoc. Comput. Linguistics*, 2003, pp. 8–15.
[7] K. Hacioglu and W. Ward, "Target word detection and semantic role chunking using support vector machines," in *Proc. Conf. North Am. Chapter Assoc. Comput. Linguistics Human Language Technol.*, 2003, pp. 25–27.
[8] S. Pradhan, K. Hacioglu, V. Krugler, W. Ward, J. H. Martin, and D. Jurafsky, "Support vector learning for semantic argument classification," *Mach. Learning*, vol. 60, no. 1, pp. 11–39, 2005.
[9] T. D. Breaux and A. I. Antón, "Deriving semantic models from privacy policies," in *Proc. 6th IEEE Int. Workshop Policies Distrib. Syst. Netw.*, 2005, pp. 67–76.
[10] T. D. Breaux and A. I. Antón, "Analyzing goal semantics for rights, permissions, and obligations," in *Proc. 13th IEEE Int. Conf. Requirements Eng.*, 2005, pp. 177–186.
[11] T. Breaux and A. Antón, "Analyzing regulatory rules for privacy and security requirements," *IEEE Trans. Softw. Eng.*, vol. 34, no. 1, pp. 5–20, Jan./Feb. 2008.
[12] T. D. Breaux, Legal requirements acquisition for the specification of legally compliant information systems. North Carolina State University, 2009.
[13] W. Ammar, S. Wilson, N. Sadeh, and N. A. Smith, "Automatic categorization of privacy policies: A pilot study," School of Computer Science, Language Technology Institute, Tech. Rep. CMU-LTI-12-019, 2012.
[14] Terms of service, didn't read project. (2016). [Online]. Available: http://tosdr.org/
[15] R. Schwitter, "Controlled natural languages for knowledge representation," in *Proc. 23rd Int. Conf. Comput. Linguistics*, 2010, pp. 1113–1121.
[16] C. A. Brodie, C.-M. Karat, and J. Karat, "An empirical study of natural language parsing of privacy policy rules using the sparcle policy workbench," in *Proc. 2nd Symp. Usable Privacy Secur.*, 2006, pp. 8–19.
[17] K. Vaniea, C.-M. Karat, J. B. Gross, J. Karat, and C. Brodie, "Evaluating assistance of natural language policy authoring," in *Proc. 4th Symp. Usable Privacy Secur.*, 2008, pp. 65–73.
[18] Privacy policy profile of xacml v3.0," 2010. [Online]. Available: http://docs.oasis-open.org/xacml/3.0/xacml-3.0-privacy-v1-spec-cs-01-en .pdf
[19] P. Inglesant, M. A. Sasse, D. Chadwick, and L. L. Shi, "Expressions of expertness: The virtuous circle of natural language for access control policy specification," in *Proc. 4th Symp. Usable Privacy Secur.*, 2008, pp. 77–88.
[20] L. Shi and D. W. Chadwick, "A controlled natural language interface for authoring access control policies," in *Proc. ACM Symp. Appl. Comput.*, 2011, pp. 1524–1530.
[21] E. B. Fernandez and J. Hawkins, "Determining role rights from use cases," in *Proc. 2nd ACM Workshop Role-Based Access Control*, 1997, pp. 121–125.
[22] P.-J. Fontaine, A. Van Lamsweerde, E. Letier, and R. Darimont, "Goal-oriented elaboration of security requirements," Universite Catholique de Louvain, Belgium, 2001.
[23] Q. He and A. I. Antón, "Requirements-based access control analysis and policy specification (ReCAPS)," *Inf. Softw. Technol.*, vol. 51, no. 6, pp. 993–1009, 2009.
[24] X. Xiao, A. Paradkar, S. Thummalapenta, and T. Xie, "Automated extraction of security policies from natural-language software documents," in *Proc. ACM SIGSOFT 20th Int. Symp. Found. Softw. Eng.*, 2012, Art. no. 12.
[25] J. Slankas, X. Xiao, L. Williams, and T. Xie, "Relation extraction for inferring access control rules from natural language artifacts," in *Proc. 30th Annu. Comput. Secur. Appl. Conf.*, 2014, pp. 366–375.
[26] C. J. Fillmore, "Frame semantics and the nature of language," *Ann. New York Academy Sci.*, vol. 280, no. 1, pp. 20–32, 1976.
[27] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *J. Mach. Learning Res.*, vol. 12, no. Aug, pp. 2493–2537, 2011.
[28] T. Barnickel, J. Weston, R. Collobert, H.-W. Mewes, and V. Stümpflen, "Large scale application of neural network based semantic role labeling for automated relation extraction from biomedical texts," *PLoS One*, vol. 4, no. 7, 2009, Art. no. e6393.
[29] M. Surdeanu and J. Turmo, "Semantic role labeling using complete syntactic analysis," in *Proc. 9th Conf. Comput. Natural Language Learning*, 2005, pp. 221–224.
[30] E. Charniak, "A maximum-entropy-inspired parser," in *Proc. 1st North Am. Chapter Assoc. Comput. Linguistics Conf.*, 2000, pp. 132–139.
[31] X. Carreras and L. Màrquez, "Introduction to the CoNLL-2005 shared task: Semantic role labeling," in *Proc. 9th Conf. Comput. Natural Language Learning*, 2005, pp. 152–164.
[32] M. Lewis, L. He, and L. Zettlemoyer, "Joint a* CCG parsing and semantic role labelling," in *Proc. Conf. Empirical Methods Natural Language Process.*, 2015, pp. 1444–1454.
[33] M. Steedman, *The Syntactic Process*. Cambridge, MA, USA: MIT Press, 2000.
[34] A. Björkelund, L. Hafdell, and P. Nugues, "Multilingual semantic role labeling," in *Proc. 13th Conf. Comput. Natural Language Learning*, 2009, pp. 43–48.
[35] X. Zhu and A. B. Goldberg, "Introduction to semi-supervised learning," *Synthesis Lectures Artif. Intell. Mach. Learning*, vol. 3, no. 1, pp. 1–130, 2009.
[36] S. He and D. Gildea, "Self-training and co-training for semantic role labeling: Primary report," Univ. Rochester, Rochester, NY, Tech. Rep. 891, 2006.
[37] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, "The stanford CoreNLP natural language processing toolkit," in *Proc. 52nd Annu. Meet. Assoc. Comput. Linguistics*, 2014, pp. 55–60.
[38] E. Charniak and M. Elsner, "EM works for pronoun anaphora resolution," in *Proc. 12th Conf. Eur. Chapter Assoc. Comput. Linguistics*, 2009, pp. 148–156.
[39] A. Meneely, B. Smith, and L. Williams, "Appendix B: iTrust electronic health care system case study," *Softw. Syst. Traceability*, vol. 2, p. 425, 2012.
[40] R. Van De Stadt, "Cyberchair: A web-based groupware application to facilitate the paper reviewing process," *CoRR*, vol. abs/1206.1833, http://arxiv.org/abs/1206.1833, 2012, .

Masoud Narouei received the BE degree in computer engineering from Shiraz University while being an active member of APA Institute of Shiraz University, Shiraz, Iran. He is currently working toward the PhD degree in computer science and engineering at the University of North Texas, Denton, Texas. He is mainly focused on developing new access control models. Contact him at masoudnarouei@my.unt.edu.

Hassan Takabi is an assistant professor of computer science and engineering with the University of North Texas, Denton, Texas. He is director and founder of the Information Security and Privacy: Interdisciplinary Research and Education (INSPIRE) Lab and a member of the Center for Information and Computer Security (CICS), which is designated as National Center for Academic Excellence in Information Assurance Research (CAE-R) and Education (CAE-IAE). His research is focused on various aspects of cybersecurity and privacy including advanced access control models, insider threats, cloud computing security, mobile privacy, privacy and security of online social networks, and usable security and privacy. He is member of the IEEE and ACM. Contact him at takabi@unt.edu.

Rodney Nielsen received a dual PhD degrees in computer science and cognitive science from the University of Colorado, Boulder, in 2008. He is currently an associate professor of computer science and engineering with the UNT, where he directs the Human Intelligence and Language Technologies (HILT) Lab with Eduardo Blanco. Prior to coming to UNT, he was an assistant professor Adjunct of Computer Science, CU Boulder, a research scientist with the CU's Center for Computational Language and Education Research, and a research scientist with Boulder Language Technologies. Contact him at rodney.nielsen@unt.edu.