Ontology-based Classification of Email

Kazem Taghva, Julie Borsack, Jeffrey Coombs, Allen Condit, Steve Lumos, and Tom Nartker

Information Science Research Institute
University of Nevada, Las Vegas

Abstract

We report on the construction of an ontology that applies rules for identification of features to be used for email classification. The associated probabilities for these features are then calculated from the training set of emails and used as a part of the feature vectors for an underlying Bayesian classifier.

1. Introduction

One of the recent challenges for the electronic record keeping community is the creation of a well-defined process to address the issues associated with legal discovery. In particular, a record manager must make a decision on the retention of certain records. For example, the Department of Energy (DOE) under Federal Regulation 10CFR2 Subpart J, must submit all records that could be of interest to the licensing proceedings of a high level nuclear waste repository (any record falling under topical guidelines given in the Nuclear Regulatory Commission's Regulatory Guide 3.69)[2]. Other examples include the retention of patient records for activities related to Occupational Medicine[11]. In a broader sense, some email messages are also considered records and must be retained under certain guidelines. Without other means of satisfying retention requirements, some of these agencies have archived all of their email. In the case of DOE, their archive is several tera-bytes in size.

The Information Science Research Institute (ISRI) at the University of Nevada, Las Vegas has been studying and building the necessary tools to classify this archive. In this paper, we report on a part of this activity which involves the building of a knowledge base to represent and process the structured data associated with the email.

This paper is divided into Five Sections. Section One is the introduction followed by Sections Two and Three on ontology and rule building. Section Four explains the process of fact generation from the individual emails for rule application. Section Five is the conclusion and future work.

2 Building the Ontology

Ontologies are applied to a myriad of tasks to formalize information within specific domains. There are several reasons for developing ontologies as pointed out by Noy and McGuinness[9]:

- To share domain information among people and software.
- To enable reuse of domain knowledge.
- To analyze domain knowledge and make it more explicit.
- To separate domain knowledge from its implementation.

Each of these reasons played a role in our decision to implement the *email ontology* but most specifically, our objective was to share in a formal way domain information with our Bayesian email classifier, *Ecdysis*. It was important that Ecdysis have access to the domain knowledge and be able to apply it for classification without coding it directly into the system.

Our email classification ontology embodies several *concepts*. These concepts were established in one of two ways: either they were aspects of the criteria prescribed by DOE for inclusionary or exclusionary records[10] or the concepts were observed in the training data. Figure 1 traces concepts included in the ontology to their source.

We implement our ontology using Protégé-2000, beta version 1.8[8]. Protégé-2000 is a knowledge-base-editing environment developed at Standford University. Protégé-2000 and other frame-based systems describe ontologies declaratively which means they explicitly

DOE Prescribed Criteria All DOE guidance and instructions to the field offices, other agencies, contractors, and national laboratories performing Office of Civilian Radioactive Waste Management (OCRWM) work are inclusionary.

Observed Concepts:

- Organization
- Department
- Email Agent
- Message Topics

Observed Training Data Concepts Emails with fewer than 40 body words and no attachments is more likely exclusionary.

Observed Concepts:

- Email Characteristics
- Count Characteristics
- Attachment Type Characteristics

Figure 1. Identifying Ontology Concepts

state the class hierarchy and the classes to which individuals belong; frames are the "building blocks" of the knowledge base.

The concepts of an ontology typically translate to *classes*. So the concepts identified in the rules noted above were defined as classes in the email ontology. *Slots* describe properties of classes, and *facets* describe properties of slots. Figure 2 shows the Protègè interface where these elements are defined.

We begin the development of the email ontology by defining its purpose, domain, and scope.

Purpose The purpose of our ontology should be to validate specific email characteristics. In Ecdysis, these characteristics become *features* and contribute to aggregate probabilities.

Domain Our domain encompasses email meta-data, message body content, inclusionary/exclusionary properties and criteria. The ontology should be flexible enough so that additional concepts from this domain can be added without extensive revision.

Scope The scope of our ontology should be limited to information that will aid us in determining whether an email should be classified as inclusionary or exclusionary. The emails we plan to classify are managed in the Lotus Notes system so there is a multitude of fields to which we have access. For example, we have access to the routing servers but since this information is of no consequence to inclusionary/exclusionary classification, it is outside the scope of our ontology.

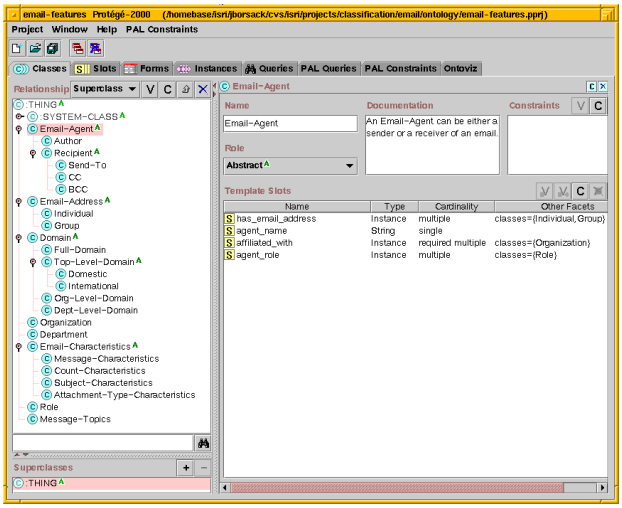

Figure 2. Defining Classes, Slots, and Facets in Protégé-2000

We used a *combination* development method by applying both top-down and bottom-up processes to specify the ontology. This approach seemed most appropriate because parts of the ontology were explicit and well-defined at the beginning while other concepts were more vague. The three prominent taxonomic hierarchies in the email ontology are email agent, email characteristics, and email domain. Each includes instances and slot values that subsequently resolve to variables in CLIPS rules (discussed in Section 3).

3 Building the Rules

Although Protégé-2000 has the capacity to store an ontology in Resource Description Framework (RDF) format[7], by default it stores an ontology as a text file compatible with the expert system shell CLIPS. CLIPS, the C Language Integrated Production System, was designed by the Johnson Space Center at NASA to provide a low-cost, portable rule-based programming system. Since version 5.0, CLIPS has also included procedural and object oriented programming elements[5].

Because an ontology contains classes and instances, Protégé-2000 stores these in CLIPS Object Oriented Language (COOL) format.[3] Protégé-2000, however, is not fully compatible with COOL because Protégé-2000

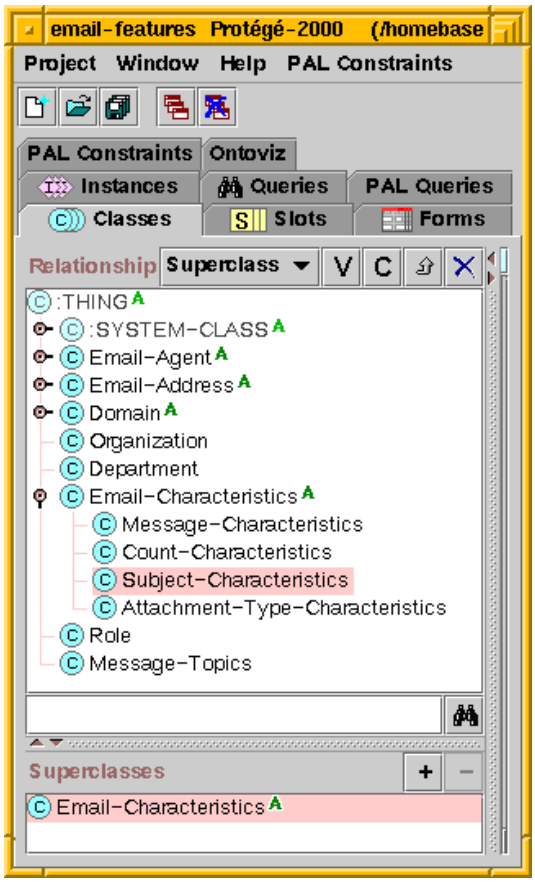

Figure 3. "Email-Characteristics" and its Subclasses

allows metaclasses and defines all slots globally while a class slot in COOL has class scope. In COOL, two classes may have the same slot name, but in Protégé-2000, every slot name must be unique.

However, if one carefully follows CLIPS and Protégé naming guidelines, no serious problems will result. In fact, we were able to exploit Protégé-2000's global slots. In CLIPS, rules in conditional ("if...then—") form are constructed. Data, called "facts" are matched with the antecedent of a rule, which "fires" if a fact fits the format of the antecedent. In addition to the classes defined by our ontology, we defined a class for the emails, and each email was defined as an instance of this class. Since Protégé-2000 had global slots, we were able to connect the slot name of an email with its corresponding slot in Protégé-2000. Although Protégé-2000 would consider such slots to be one and the same, CLIPS happily viewed them as distinct slots belonging to distinct classes.

As a result, rules discovered from the training data or prescribed by the DOE could be written as CLIPS rules. These rules typically followed one basic format. One or more features of an email instance would be matched with instances of classes from the ontology. If there was a successful match, then the rule would fire causing a message to be printed stating that a given email was an instance of a concept.

For example, consider an email e with the subject line *Lunch*. Our review of sample emails led us to conclude that such emails are very unlikely candidates for inclusion in the permanent archive. In our ontology this information was stored as an instance of a class called `Subject-Characteristics` which is a subclass of `Email-Characteristics`. `Email-Characteristics` is the superclass of classes which describe various aspects of an email which we considered significant to our classification task. A Protégé view of this class appears in Figure 3. Besides `Subject-Characteristics` which describes characteristics of the subject line of emails, other subclasses of `Email-Characteristics` are `Message-Characteristics`, `Count-Characteristics`, and `Attachment-Type-Characteristics`.

The subclass `Count-Characteristics` describes numerical information about an email which was determined relevant to whether it should be included or not. For example, we noted that if the number of unique terms in an email was less than 40, then it would be more likely that the email should not be included in the archive. `Attachment-Type-Characteristics` contains information about documents attached to the email which might contribute to the inclusionary status of an email. `Message-Characteristics` considers more general aspects of an email, such as its type. For example, if an email is a proposal, then it would more likely need to be included in the repository.

The CLIPS rule for our lunch example will first determine what the subject line of the email is. It will next compare this subject line with instances of `Subject-Characteristics` to determine if there is a match. Each instance of the class `Subject-Characteristics` also has a slot `incl_excl` which contains `E` if the subject line tends to make the email exclusionary. *Lunch* is such an exclusionary subject line. Other instances of `Subject-Characteristics` are shown in Figure 4 as they appear in Protégé.

Once it is determined that email e has the exclusionary subject line *Lunch*, the CLIPS rule will "fire" and output that email e has the feature of containing an exclusionary subject line. This feature becomes one of many that can now be used for training and classification within our Bayesian classifier Ecdysis.

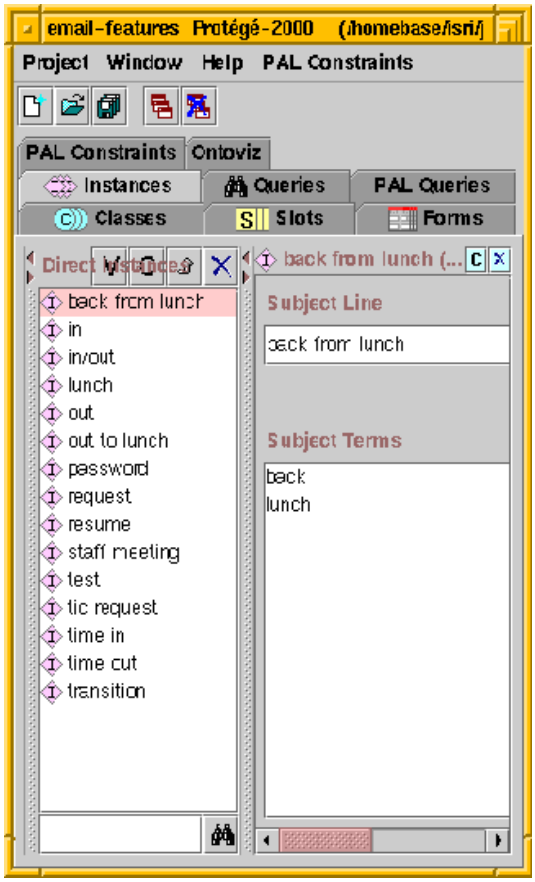

Figure 4. Instances of the Class "Subject-Characteristics"

```
([S000046947-email_author-1] of email_author
 (agent_name           "Kathryn Cooper")
 (top_domain_name      "gov")
 (individual_username  "kathryn_cooper")
 (org_domain_name      "ymp")
 (dept_domain_name     "ym"))

([S000046947-email_send_to-1] of email_send_to
 (agent_name           "Nick Connerley")
 (top_domain_name      "gov")
 (individual_username  "nick_connerley")
 (org_domain_name      "ymp")
 (dept_domain_name     "ym"))

([S000046947-email] of email
 (id                  S000046947)
 (body_lengths        9)
 (attachment_counts   0)
 (subject_line        "Lunch")
 (email_author        [S000046947-email_author-1])
 (email_send_to       [S000046947-email_send_to-1]))
```

Figure 5. An Email Converted to CLIPS Instance

One alternative we considered to using CLIPS is the OntoJava[4] system which combines RDF files created by Protégé with the rule mark-up language MDL[1]. We opted for CLIPS at least in the prototyping stage because we believe that CLIPS is a more mature and stable programming system.

4 The Working System

The emails were originally managed within a Lotus Notes system. Lotus Notes is able to convert emails into extended markup language (XML) using a format called Domino extended markup language (DXL) [6]. To transform these DXL documents into a format understandable to CLIPS, a program was written to take information from various fields in the DXL document and write them to slots within a CLIPS class instance. An example of the output of the program appears in Figure 5.

Three instances are shown in Figure 5. Strings in square brackets, such as `[S000046947-email]` are *instance names* in CLIPS. Hence `[S000046947-email]` is an instance of a class `email` whose slot `id` takes the value `S000046947`, the slot `body_lengths` has the value `9`, and so on. Since each email has at least one author and perhaps several recipients, an instance of an email must be related to the author and recipients in some way. Two classes called `email_author` and `email_send_to` were defined and `[S000046947-email_author-1]` and `[S000046947-email_send_to-1]` are respective instances of each. The email instance `[S000046947-email]` is connected to its author and recipient via its slots `email_author` and `email_send_to` which take *instance names* as their values.

Once the email is converted into a collection of CLIPS instances, it can be loaded into a CLIPS program along with class definitions and instances generated by Protégé. The rules will match email instances with class instances from the ontology and if all predicates in the left hand side are satisfied, the CLIPS rules will fire. The final output from CLIPS will list the features discovered for each email. These features are then given to our Bayesian classifier which along with other features, can help train the classifier or categorize test emails. A representation of the system appears in Figure 6.

5 Conclusion and Future Work

Determining if a record, in this case an email, should be retained for legal discovery requires more than pure text categorization. The requirement for 100% precision allows no room for error. By leveraging structured meta

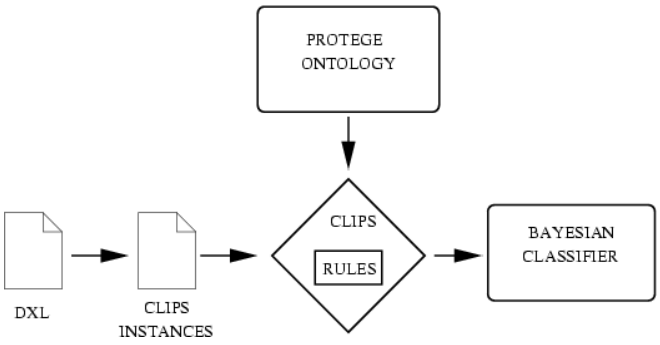

Figure 6. The System

data and formalizing logical rules with an ontology, we provide a more complete solution. Knowledge about who sent or received an email, what organization they work for, and what position they hold in an organization cannot typically be deduced from the individual email alone. However, such information added to our ontology which, when combined with our rules, allows us to enhance our Bayesian categorizer, Ecdysis. Preliminary tests show that these additional features enhance the accuracy of our classification system dramatically.

We have also discovered that this methodology can be generally applied to other classification challenges. For example, DOE must filter all publicly available documents for potentially sensitive information (PSI). Although differences exist between the two problems, ISRI will apply a similar strategy to its solution.

References

[1] H. Boley. The rule markup initiative. `http://www.dfki.uni-kl.de/ruleml`.

[2] N. R. Commission. Regulatory guide 3.69. http://www.nrc.gov/NRC/RG/03/03-069.html, 1996.

[3] C. Culbert, G. Riley, and B. Donnell. *CLIPS Reference Manual*, 2002. `http://www.ghg.net/clips/download/documentation`.

[4] A. Eberhart. OntoJava. `http://www.i-u.de/schools/eberhart/ontojava`.

[5] J. Giarratano and G. Riley. *Expert Systems: Principles and Programming*. ITP, 3rd edition, 1998.

[6] IBM. Dxl resources. `http://www-10.lotus.com/ldd/dxl`.

[7] E. Miller, R. Swick, and D. Brickley. Resource description framework (RDF). `http://www.w3.org/RDF`.

[8] N. Noy and D. L. McGuinness. The knowledge model of protege-2000: Combining interoperability and flexibility. In *2th International Conference on Knowledge Engineering and Knowledge Management (EKAW'2000)*, Juan-les-Pins, France, 2000.

[9] N. Noy and D. L. McGuinness. Ontology development 101: A guide to creating your first ontology. 2001. Technical Report SMI-2001-0880, Standford Medical Informatics, Stanford University, Stanford, CA 94305, 2001.

[10] Office of Civilian Radioactive Waste Management. Programs records management: Receipt and handling of program records and records packages, September 1992. Accession Number: HQF.930510.0048.

[11] K. Taghva, J. Borsack, T. Nartker, S. Lumos, and A. Condit. Managing occupational medicine historical data. In *Proceedings of the 2002 International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences (METMBS'02)*, 2002.