Automating the Generation of Semantic Annotation Tools Using Clustering Techniques

Vitor Souza[1], Nicola Zeni[1], Nadzeya Kiyavitskaya[1], Periklis Andritsos[1], Luisa Mich[2], and John Mylopoulos[1]

[1] Dept. of Information Engineering and Computer Science
[2] Dept. of Computer and Management Sciences,
University of Trento, Italy

Abstract. In order to generate semantic annotations for a given collection of documents, one needs an annotation schema consisting of a semantic model (a.k.a. ontology) along with lists of linguistic indicators (keywords and patterns) for each concept in the ontology. The focus of this paper is the automatic generation of the linguistic indicators for a given annotation schema and a corpus of documents. Our proposal needs only a small number of user-defined seeds and bootstraps itself by exploiting novel clustering techniques. The baseline for this work is the Cerno project [14] and the clustering algorithm LIMBO [2]. Along with the proposed technique, we present results that compare the output of the clustering algorithm with linguistic indicators created manually for two case studies.

1 Introduction

The process of enriching fragments of a document with metadata describing their meaning – known as *semantic annotation* – is commonly recognized as the one of the cornerstones of the Semantic Web [5]. Accordingly, to generate domain-dependent metadata any semantic annotation system utilizes a semantic model, a.k.a. ontology, that covers a domain of interest at a suitable level of granularity. This model is part of an *annotation schema* along with sets of linguistic indicators (keywords and patterns) that identify instances of each concept in the model. These indicators determine what text fragments are to be annotated, and are usually constructed manually by a domain expert.

In this work we explore the applicability of some of the main ingredients of a supervised categorical clustering algorithm LIMBO [4] for producing linguistic indicators for a given semantic model. LIMBO was originally proposed for clustering structural information of database tuples in relational databases. The main motivation for applying this method is that it does not require large amounts of training data to bootstrap the learning algorithm and needs a limited amount of human attention at the initial stage. Moreover, the distance measure employed in LIMBO works with categorical data (i.e. data that do not have an inherent order) unlike most clustering techniques. Our primary goal in this work is to verify whether such a lightweight approach can facilitate the construction of an annotation schema, given a semantic model and a training set of documents.

As the basis for conducting experiments on semi-automatic generation of the annotation schema we refer to the architecture of our semantic annotation system called Cerno [14]. Cerno accepts as input a grammar and a text document, generates a parse tree for the input document, and applies transformation rules to generate output in a target format. The approach discriminates between domain-dependent and independent components of the annotation process and thus allows for easy adaptation to different application domains and tasks. However, an initial effort is needed to adapt Cerno to a specific domain.

In Cerno, semantic tags are inferred on the basis of a domain-dependent annotation schema. This schema contains a list of tags for concepts to be identified, selected from the domain semantic model, and a vocabulary of indicators related to each concept. Indicators can be literal words and phrases, or names of parsed entities. Domain-specific indicators can be derived manually, i.e. proposed by the domain experts, or semi-automatically, for instance, mined from a rich conceptual model representing the domain knowledge (if such a model is available). Accordingly, the focus of this paper is automation of the generation of an annotation schema for a given semantic domain using a clustering approach. We evaluate the performance of the method on two different data sets and the related annotation schemas borrowed from the previous applications of Cerno.

This paper is structured as follows. The baseline of the present work in sketched in Section 2. It introduces the semantic annotation framework Cerno and LIMBO, the clustering technique adopted. Section 3 describes how the baseline technologies were extended and shows the tool built on top of the LIMBO. Section 4 presents the setup and evaluation of two experimental case studies and summarizes the lessons learned. Section 5 recalls the related work. Finally, the conclusions are drawn in Section 6.

2 Research Baseline

2.1 Cerno semantic annotation framework

Cerno is a lightweight semantic annotation framework that exploits fast and scalable techniques from the software reverse engineering area, more specifically, "design recovery" process [10]. To annotate input documents, Cerno uses context-free grammars, generates a parse tree, and applies transformation rules to generate output in a target format [13]. The reader can find a detailed description of the architecture and the performance of the system in [14].

Normally, adapting Cerno to a new application domain requires a couple of weeks, because its domain dependent components should be tuned for a given type of document and a specific annotation schema. In particular, this process requires the manual construction of indicators to identify semantic concepts, on the basis of which it provides automated assistance to engineers. Each indicator defines a pattern or contextual keywords that characterizes instances of a concept. Accordingly, in this work we explore the possibilities to automate generation of such indicators for specific semantic domain. Having a set of examples, one can try to identify a set of contextual keywords describing relevant concepts using well-established statistical methods that have been proven effective

in many areas. More specifically, we consider clustering methods as a suitable approach to deal with the problem. These techniques aim at finding similar text fragments and therefore can be a useful means in providing an approximate set of indicators for semantic concepts. To this end, we've been experimenting with a scalable hierarchical categorical clustering algorithm called LIMBO [2], [4].

There are several semantic domains where Cerno's framework has demonstrated good performance in terms of quality of the produced annotations and time savings. In particular, the best results were achieved in semantic annotation of text documents in the domains of announcements for accommodation taken from on-line newspapers [15] and identification of rights and obligations in the HIPAA privacy rule [16]. These two data sets Cerno were reused in the present work as two case studies.

2.2 Data Clustering with LIMBO

Data clustering [12] is a common technique for statistical data analysis and is widely used in many fields. Clustering is the process of generating collections of similar text documents. Accordingly, the clustering task is, given a source of textual documents and similarity measure, find several clusters of documents that are relevant to each other. Clustering techniques can be divided into two main groups: partitioning methods and hierarchical methods. Hierarchical algorithms find successive clusters using previously established clusters, whereas partitioning algorithms determine all clusters at once.

Our approach is inspired by LIMBO [2], a scalable hierarchical categorical clustering algorithm that builds on the Information Bottleneck (IB) [18] framework for quantifying the relevant information preserved when clustering. The algorithm proceeds in three phases: Phase 1 constructs a cluster representative for the initial data set for efficiency purposes, Phase 2 performs the clustering on the representative and Phase 3 labels the initial input with the appropriate cluster information.

In our work we assume that apart from the initial data set, we give the algorithm as input an initial clustering. This clustering corresponds to the set of input records that contain the keywords a user indicated as seeds for the underlying semantic domain. As a consequence, we shall process the data in a hierarchical fashion, starting with the initial clustering of the documents and proceeding until all relevant documents have been identified.

The method proceeds as follows: (1) Given a clustering C, group the set of input documents T into the corresponding clusters S_t; (2) Merge all documents of S_t into a representative document R_t; (3) Find the documents of $S \backslash \mathrm{S}_t$ that are closest to the representative R_t; (4) Analyze the documents found in step 3 for new semantic annotations of the domain and add them to S_t; (5) Repeat from step 2 until stopping criteria are satisfied.

We describe the way in which new annotations of the domain are discovered and the stopping criteria in Section 3. Two important aspects of the previous procedure are the distance measure used to decide which documents are close to the document representatives as well as the way in which documents are merged into their representative. First, we examine the data representation we assume.

We will assume that the input to our problem is a set $\mathbf{T}$ of n text documents defined on the set of values (keywords) V, with domain $\mathbf{V}$. Document t_i contains d_i values. If d is the size of the domain $\mathbf{V}$, we can represent our data as an $n \times d$ matrix M, where each $t \in \mathbf{T}$ is a d-dimensional row vector in M. If tuple t contains attribute value v, then $M[t, v] = 1$, otherwise $M[t, v] = 0$.

Now, let T and V be random variables that range over sets $\mathbf{T}$ and $\mathbf{V}$ respectively. For document $t_i \in \mathbf{T}$, $1 \leq i \leq n$ we define

$\mathrm{p}(\mathrm{t}_i) = 1/n$

$p(v|t_i) = \begin{cases} 1/d_i \text{ if } v \text{ appears in } t \\ 0 \quad \text{otherwise} \end{cases}$.

Given an input clustering, our task now is to build the cluster representatives, against which each document will be compared. We generate a representative for the cluster of documents in a *Distributional Cluster Feature (DCF)* [2]. We will use the information in the relevant *DCF*s to compute the distance between a cluster representative and a document. Let $\mathbf{T}$ denote a set of documents over a set $\mathbf{V}$ of values, and let T and V be the corresponding random variables, as described earlier. Also let $\mathbf{C}$ denote a clustering of the documents in $\mathbf{T}$ and let C be the corresponding random variable. The *Distributional Cluster Feature* (DCF) of a cluster with identifier c_i is defined by the pair

$$DCF(c_i) = \Big(|c_i|, p(V|c_i)\Big)$$

where $|c_i|$ is the cardinality of cluster c_i, and $p(V|c_i)$ is the conditional probability distribution of the values given the cluster c_i.Practically, the cluster representative of rep_i of cluster c_i will be its DCF, $DCF(c_i)$.

If c_i consists of a single tuple, then $|c_i| = 1$, and $p(V|c_i)$ is computed as described in the previous subsection. For larger clusters, the DCF is computed recursively as follows. Let c^* denote the cluster we obtain by merging two clusters with identifiers c_1 and c_2. The DCF of the cluster c^* is equal to $DCF(c^*) = \Big(|c^*|), p(V|c^*)\Big)$, where $|c^*|$ and $p(V|c^*)$ are computed using the following equations:

$|c^*| = |c_1| + |c_2|$

$p(V|c^*) = \frac{|c_1|}{|c_1|+|c_2|} p(V|c_1) + \frac{|c_2|}{|c_1|+|c_2|} p(V|c_2)$

Intuitively, when computing a new cluster representative, its cardinality becomes the sum of the cardinalities of the merged sub-clusters, while the conditional probability of its values is the weighted average of the conditional probability distribution of the values from the merged sub-clusters. Note that a cluster representative may not necessarily belong to the initial set of documents.

Our notion of distance d between documents and cluster representatives that include categorical values, is based on an intuitive calculation of the loss of information when two distributions are merged. Information here is defined in terms of Information Theory [9]. More precisely, we use the *mutual information* $I(X;Y)$ of two random variables X and Y, to quantify the information that variable X contains about Y and vice versa. To compute the value of $I(X;Y)$, we need the probability distributions of X and Y. In our case, we quantify the information that the clusters in C contain about the values in V and use the

distributions that describe the documents and clusters stored in the corresponding *DCF*s. The main objective of the quantification of the loss of information is to realize which documents share as many common keywords as possible with their cluster representative. Thus, given the *DCF*s of representatives s_1 and s_2, the information loss (and hence the distance) between s_1 and s_2 is given by the following expression:

$$\mathrm{d}(\mathrm{s}_1, s_2) = I(C;V) - I(C';V)$$

where C' denotes the clustering after merging the representatives.

In the following section we show how the LIMBO algorithm was integrated in a user-friendly tool, in this way we address the need in generating linguistic indicators for Cerno's semantic annotation process.

3 Architecture

Given that Cerno factors out domain-dependent knowledge into separate components, the framework can serve as the basis for the generation of new semantic annotation tools. To provide a user assistance in constructing these components, we propose to apply the clustering technique LIMBO. Having such a support will allow to quickly adapt the framework to new application domains in terms of both different annotation schemas and types of documents. Fig. 1 shows the new LIMBO-based process.

To realize this goal and verify effectiveness of clustering techniques for generating linguistic indicators for Cerno, we have implemented a tool based on the LIMBO algorithm. This tool has a graphical user interface (GUI) developed in Java. According to the algorithm described in Section 2.2 the input to LIMBO includes: the initial data set that is then transformed into a cluster representative by the algorithm and a set of documents which are used for training. Thus, the tool accepts non annotated textual documents as the training set. The initial clustering is represented by a handful of seed words specified by the user for the underlying semantic domain. They corresponds to the set of input records that contain these words. This graphic interface provides a step-by-step wizard that allows the user to configure the experiment, then separates the input file into clusters, repeatedly runs the algorithm and provides the results of each run. Figure 2 shows the tool's configuration screen.

To initialized LIMBO, on the first step of the wizard the user specifies the following input files and parameters:

- *The input document* is the original unannotated input document.
- *The clusters file* is the text file that contains the clustering information, i.e. the keyword seeds together with the semantic concept they belong to.
- *The stopping criterion* specifies the way of terminating the algorithm, it can be defined as a fixed number of iterations or as a lower threshold. Threshold is a number defined by the user and characterizes the distance between the cluster representative and the input document. This means that the smaller the distance is, the finer are the results produced by the algorithm. In cases where the user defines the stopping criterion through the threshold, the algorithm is run until no cluster surpasses the threshold.

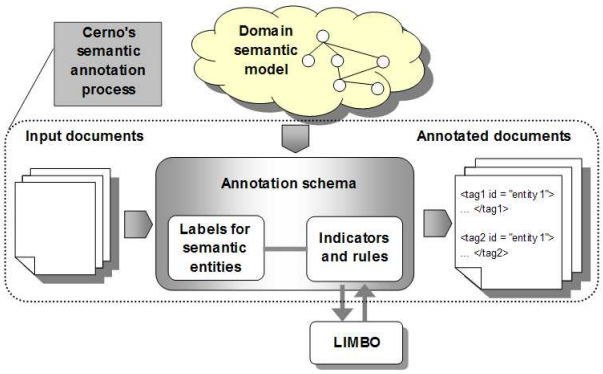

Fig. 1. The LIMBO module in the semantic annotation process

Fig. 2. Configuration screen for the LIMBO-based tool.

- *The parsing mode* defines how the clusters will be generated from the input document, i.e. if the input file will be separated by sentences, by all punctuation marks or n-grams. An n-gram is a sub-sequence of n items from a given sequence. In our case, we take n words starting at the 1^{st} word of the sentence, then n words starting at the 2^{nd} word, and so forth. Thus, the n-grams parsing mode generates the largest number of clusters and consequently requires longer processing times.
- *The analyzer* is the module responsible for extracting the keywords out of the input document. Currently, the prototype contains two standard analyzers for English language: (a) standard analyzer that performs basic analysis of the text, setting all letters to lowercase and removing stopwords and punctuation marks; (b) standard analyzer with stemming that in addition to the former one applies the process of reducing inflected or derived words to their stem, base or root form (e.g. "player" and "playing" become "play"). Other analyzers can be developed and easily integrated into the prototype.

In the next step, the tool parses the input file as specified in the configuration and separates the clusters using the specified parsing mode. Then, for each cate-

gory, it marks the clusters that contain any of the keywords of the category and runs the algorithm as many times as specified in the stopping criterion. When ran a fixed number of times, the k nearest neighbors are marked at each run; when ran with a threshold, all clusters that are below the threshold are included. When finished, the prototype shows for each category the most relevant words in each run of the algorithm.

4 Experimental Case Studies

To verify feasibility of the proposed approach, we applied the LIMBO-based tool on two different experiments. In this study we compare the output of the clustering algorithm with linguistic indicators created manually for two data sets used in previous applications of Cerno: the *classified accommodation ads* drawn from an on-line newspaper and the *U.S. privacy rule HIPAA*. In both semantic domains, Cerno has demonstrated good quality of the annotation results given the linguistic indicators derived manually [15], [16].

The stopping criterion for the clustering algorithm was set to 10 iterations with the addition of the 2 nearest-neighbors. We selected 10 as the number of iterations empirically, given that the output of the clustering algorithm remained almost unchanged after a number of iterations greater than 10. The tool was run with 8 different parsing configurations per each experiment:

- Run 1: sentences parsing mode without stemming,
- Run 2: parsing mode for all punctuation marks without stemming,
- Run 3: 3-grams parsing mode without stemming,
- Run 4: 7-grams parsing mode without stemming,
- Run 5: sentences parsing mode with stemming,
- Run 6: parsing mode for all punctuation marks with stemming,
- Run 7: 3-grams parsing mode with stemming,
- Run 8: 7-grams parsing mode with stemming,

Number 3 for n-grams mode was chosen to account for commonly used word collocations, such as for instance "information system" or "health care cleaninghouse", and number 7 was defined as the highest upper bound for a possible number of words in collocations.

4.1 Evaluation Method

We evaluated the performance by comparing automated results to a *Gold model*, i.e. the list of indicators drawn manually by the experts, and calculating recall and precision quality measures [20]. Let TP be the number of true positives, i.e. relevant items retrieved, FP – the number of false positives, i.e. irrelevant items retrieved, FN – the number of false negatives, i.e. relevant items missed, and $TP + FP$ – the total number of retrieved items. Then, the quality measures are defined as follows:

- *Recall* shows how well the tool performs in finding relevant items and is calculated by dividing the number of relevant items found by the number of all relevant items in the collection: $Recall = TP / (TP + FN)$;
- *Precision* shows how well the tool performs in not returning irrelevant items and is produced by dividing the number of relevant items found by the number of all items found: $Precision = TP / (TP + FP)$.

4.2 The HIPAA experiment.

The Cerno adaptation to the text of the Health Insurance Portability and Accountability Act (HIPAA) [19] recognizes document structure in terms of section and subsection boundaries, titles and annotated paragraph indices, identifies instances of the concepts actor, policy, event, information and date and annotates document fragments describing rights, anti-rights, obligations, anti-obligations, and related constraints. To generate these annotations, the tool used a list of normative phrases for the objects of concern that was obtained by manual analysis of the HIPAA document [6].

Thus, the purpose of this experiment was to evaluate how many of these indicators can be extracted by the clustering techniques. We used as input four semantic categories and several corresponding keyword-seeds, that were chosen randomly among a set of manually derived normative phrases: : *Obligation*: "must", "will"; *Right*: "may", "permits", "does not restrict"; *Exception*: "except", "apart"; *Condition*: "during", "within the time". Due to the lack of space, we show only a sample of the result produced in this experiment for the *Right* category, see Table 1. The table contains the top 20 of indicators produced by the tool given the configuration with stemming turned on and 3-grams parsing mode. The words in **boldface** appear among the manually derived indicators.

Table 1. A fragment of the result produced in the HIPAA experiment (Run 7).

Category	**Keywords**
Right	busi, condit, cover, disclos, disclosur, entiti, health, individu, inform, **may**, otherwis, paragraph, **permit**, protect, provid, **requir**, secretari, section, under, use

The evaluation was based on the list of indicators drawn manually by the experts from the HIPAA [6]. Table 2 displays the recall and precision scores for all configurations. Overall, in this experiment the tool has demonstrated low recall, except for the *Condition* concept. Better results were obtained for the runs with the stemming analyzer. Among the unstemmed results, the best average score is delivered by the 3-grams parsing mode. The processing times changes depending on parsing mode. In particular, n-grams mode causes generation of a larger number of clusters from the input document, compared to other two modes, thus increasing processing times of the algorithm.

4.3 The Accommodation Ads Experiment.

In our previous work, to annotate advertisements for accommodation in Rome drawn from an on-line newspaper, we used the annotation schema which represented the information needs of a tourist and included the concepts: *Accommodation Type, Contact, Facility, Term (of availability), Location,* and *Price*. The lists with linguistic indicators were constructed by hand from a set of examples.

This experiment utilized the same input documents and categories from an earlier experiment using accommodation ads retrieved from tourism websites

Table 2. Evaluation of the HIPAA experiment.

Run	Measure	*Condition*	*Exception*	*Obligation*	*Right*	AVG	Time in min
1	Recall	0.75	0.15	0.33	0.13	**0.34**	2.55
	Precision	0.14	0.5	0.1	0.05	**0.08**	
2	Recall	0.75	0.15	0.17	0.13	**0.30**	6.12
	Precision	0.17	0.5	0.05	0.05	**0.08**	
3	Recall	0.75	0.15	0.33	0.25	**0.37**	34.34
	Precision	0.14	0.5	0.1	0.1	**0.10**	
4	Recall	0.75	0.15	0.17	0.13	**0.30**	26.02
	Precision	0.13	0.5	0.05	0.05	**0.07**	
5	Recall	0.75	0.15	0.33	0.38	**0.40**	2.04
	Precision	0.13	0.5	0.1	0.15	**0.11**	
6	Recall	0.75	0.15	0.33	0.38	**0.40**	4.18
	Precision	0.15	0.5	0.1	0.15	**0.11**	
7	Recall	0.75	0.15	0.33	0.38	**0.40**	23.46
	Precision	0.14	0.5	0.1	0.15	**0.11**	
8	Recall	0.75	0.15	0.33	0.38	**0.40**	18.17
	Precision	0.14	0.4	0.1	0.15	**0.11**	

[15]. One third of the keywords found through the manual extraction process performed previously were included in the clusters file. The keywords were selected randomly.

Table 3 lists the keywords produced for the *Location* category. The words in **boldface** appear among the manually derived indicators. The *italicized* words are those that do not appear in the hand-crafted list, but found relevant by a human judge. These new relevant words were not counted as true positives in the evaluation results. However, one can use them to improve the annotation schema for the given application domain.

Table 3. A fragment of the result produced in the ads experiment (Run 7).

Category	Keywords
Location	apart, **center**, **close**, *distanc*, first, from, **heart**, histor, **locat**, *minut*, **near**, *piazza*, *rome*, *trastever*, *vatican*, veri, **via**, **walk**, *within*, **zone**

The annotation schema obtained in the previous application of Cerno on accommodation ads was used to evaluate the LIMBO-based technique. By counting the number of keywords found and missed by the tool, recall and precision can be calculated. Table 4 shows the corresponding figures.

This experiment has shown the results of higher quality in respect to both recall and precision values. Similarly to the first experiment, the runs with the stemming option turned off has demonstrated higher scores. In both group of 4 configurations, one with stemming turned on and the second with stemming turned off, the best average scores were obtained for the 3-grams parsing mode.

4.4 Discussion of the Results

Summarizing the evaluation results we can say that it is most effective to use the 3^{rd} and 7^{th} configurations of the algorithm, i.e. with the 3-grams parsing mode,

Table 4. Evaluation of the accommodation ads experiment.

Run	**Measure**	*Contact*	*Facility*	*Location*	*Price*	*Term*	*Type*	**AVG**	Time in min
1	Recall	0.40	0.12	0.32	0.45	0.41	0.42	**0.35**	9.55
	Precision	0.11	0.43	0.38	0.29	0.37	0.24	**0.30**	
2	Recall	0.80	0.13	0.28	0.45	0.47	0.42	**0.43**	27.29
	Precision	0.22	0.53	0.33	0.33	0.38	0.25	**0.34**	
3	Recall	0.80	0.16	0.32	0.45	0.59	0.50	**0.47**	1h33.14
	Precision	0.21	0.67	0.40	0.33	0.50	0.30	**0.40**	
4	Recall	0.80	0.16	0.28	0.45	0.59	0.42	**0.45**	1h36.12
	Precision	0.19	0.60	0.35	0.31	0.53	0.25	**0.37**	
5	Recall	0.60	0.14	0.29	0.56	0.67	0.29	**0.42**	13.29
	Precision	0.19	0.42	0.29	0.28	0.40	0.10	**0.28**	
6	Recall	0.80	0.16	0.33	0.56	0.67	0.43	**0.49**	37.32
	Precision	0.29	0.45	0.35	0.29	0.40	0.15	**0.32**	
7	Recall	0.80	0.19	0.38	0.56	0.75	0.57	**0.54**	1h51.43
	Precision	0.29	0.55	0.40	0.29	0.45	0.19	**0.36**	
8	Recall	0.80	0.19	0.33	0.56	0.67	0.43	**0.50**	1h28.24
	Precision	0.25	0.58	0.33	0.29	0.40	0.15	**0.33**	

to obtain the output of the best quality either for stemmed or non-stemmed processing. On the other hand, 3-grams parsing mode generates the largest number of clusters from the input text, thus essentially increasing processing times of the LIMBO algorithm. However, the processing time is not the most crucial criteria in the problem of generating domain-dependent indicators for a semantic annotation tool. The scarcest resource is the time spent by human experts to filter the final results.

As regards the specifics of two semantic domains, it follows that the legal documents are more difficult for automated generation of linguistic indicators using LIMBO-based technique. This shortcoming is caused by the nature of the concepts of interest. Right, obligation, condition, and exception are very abstract entities and normally span relatively large text fragments, which makes it difficult to apply clustering techniques to identify appropriate contextual keywords. While short ads documents written in a very precise language and having similar structure provide a better learning environment for the LIMBO method.

Although it may seem that the Limbo-based technique does not achieve desired high recall values, the provided evaluation results do not take into account the new relevant keywords found by the algorithm (e.g. the italicized words shown in Table 3). Therefore, we believe that LIMBO can provide a more complete approach to populate annotation schema with domain-specific indicators. Results produced by LIMBO can be a good starting point for a human expert when working with a new semantic domain. Using clustering techniques we are better able to support the generation of new semantic annotation tools in a systematic way.

To further improve the LIMBO-based tool, we plan to provide a better guidance to the user through the underlying process. For instance, we suggest to allow the user to visualize the minimum, maximum and average distances be-

tween the cluster representative and the document, so that they can properly define the value of threshold on the basis of these figures.

5 Related Work

There are several proposals for weakly supervised methods intended to populate an ontology. One of these is the *Class-Example* method [17] that exploits lexico-syntactic features to learn a classification rule from a seed set of terms. In contrast, the *Class-Pattern* approach [11] relies on using a set of patterns that indicate the presence of certain relationships, such as "is-a". *Class-Word* technique [8] uses contextual features to extract features in which a concept occurs.

Among the systems that use statistical techniques for populating semantic models is Ontosophie [7]. The system is based on machine learning natural language processing techniques to learn extraction rules for the concepts of a given ontology combining a shallow parsing tool called Marmot and a conceptual dictionary induction system called Crystal. In order to prioritize the rules learned, Ontosophie assigns a confidence value to each rule. Then, the rules obtained can be used to populate the ontology with concept instances. The OntoPop [1] methodology strives for documents annotation and ontology population under a unified framework. In addition, it adopts two other tools: Intelligent Topic Manager for representing and managing the domain model and Insight Discoverer Extractor for extracting information from texts. Firstly, the integrator runs the information extraction tool on a subset of documents, then a conceptual tree is produced for each document. Similarly to these systems, we suggest that our approach can be a cost-effective tool to support generation of extraction rules for a semantic model.

6 Conclusions and Future Work

In this work, we explore the problem of generating linguistic indicators for semantic annotation tools. The contribution of this paper consists of utilizing novel statistical clustering techniques and in particular is inspired by LIMBO [2] in order to automatically generate these indicators. Moreover, in order to allow experimenting with clustering techniques and facilitate the user's work, a tool implementing the LIMBO algorithm was developed in Java. We verified the effectiveness of the proposed technique in two different case studies.

Our future work includes further experimentation with different configurations of the clustering technique in order to improve the quality of results produced. As well, we propose to actually run semantic annotation experiments using the linguistic indicators generated by the LIMBO tool, in order to better assess their effectiveness.

Acknowledgments. This work has been partially funded by the EU Commission through the SERENITY project and by Provincia Autonoma di Trento through the STAMPS project.

References

1. Amardeilh, F.: OntoPop or how to annotate documents and populate ontologies from texts. In: Proc. of the ESWC 2006 Workshop on Mastering the Gap: From Information Extraction to Semantic Representation, Budva, Montenegro (2006)
2. Andritsos, P., Tsaparas, P., Miller, R. J., Sevcik, K. C.: LIMBO: Scalable Clustering of Categorical Data. In: Proc. of EDBT'04 (2004)
3. Andritsos, P., Fuxman, A., Miller, R. J.: Clean Answers over Dirty Databases: A Probabilistic Approach. In: Proc. of ICDE'06 (2006).
4. Andritsos, P.: Scalable clustering of categorical data and applications. PhD thesis, University of Toronto, Canada (2004)
5. Berners-Lee, T., Fischetti, M.: Weaving the Web. Harper San Francisco, chapter 12 (1999)
6. Breaux, T. D., Vail, M. W., Antón, A. I.: Towards regulatory compliance: Extracting rights and obligations to align requirements with regulations. In: Proc. of RE'06, pp. 46-55, Washington, DC, USA, IEEE Computer Society (2006)
7. Celjuska, D., Vargas-Vera, M.: Ontosophie: A Semi-Automatic System for Ontology Population from Text. In: Proc. of ICON'04, Hyderabad, India (2004)
8. Cimiano, P., Völker, J.: Towards Large-Scale, Open-Domain and Ontology-Based Named Entity Classification. In: Proceedings of RANLP'05, pp. 166–172 (2005)
9. Cover, T. M., Thomas, J.A.: Elements of Information Theory. Wiley & Sons, New York, NY, USA (1991)
10. Dean, T.R., Cordy, J.R., Schneider, K.A., Malton, A.J.: Using design recovery techniques to transform legacy systems. In: Proc. of ICSM'01, pp. 622–631 (2001)
11. Hearst, M.: Automated Discovery of WordNet Relations. In: WordNet: An Electronic Lexical Database, Christiane Fellbaum (ed.) MIT Press (1998)
12. Jardine, N., Sibson, R.: The construction of hierarchic and non-hierarchic classifications. The Computer Journal, vol. 11, pp. 117–184 (1968)
13. Kiyavitskaya, N., Zeni, N., Mich, L., Cordy, J.R., Mylopoulos, J.: Applying software analysis technology to lightweight semantic markup of document text. In: Proc. of ICAPR'05. LNCS, vol. 3686, pp. 590-600. Springer, Berlin (2005)
14. Kiyavitskaya, N., Zeni, N., Mich, L., Cordy, J. R., Mylopoulos, J.: Text mining through semi automatic semantic annotation. In: Proc. of PAKM'06. LNCS, vol. 4333, pp. 143-154. Springer-Verlag (2006)
15. Kiyavitskaya, N., Zeni, N., Mich, L., Cordy, J. R., Mylopoulos, J.: Annotating Accommodation Advertisements using CERNO. In: Proc. of ENTER'07, pp. 389–400. Springer Verlag, Wien (2007)
16. Kiyavitskaya, N., Zeni, N., Mich, L., Breaux, T. D., Antón, A. I., Mylopoulos J.: Extracting Rights and Obligations from Regulations: Towards a Tool-Supported Process. In: Proc. of ASE'07, pp. 429-432, IEEE Computer Society, ACM Press (2007)
17. Tanev, H., Magnini, B.: Weakly Supervised Approaches for Ontology Population. In: Proc. of EACL'06, Trento, Italy (2006)
18. Tishby, N., Pereira, F.C., Bialek, W.: The Information Bottleneck Method. In 37th Annual Allerton Conf. on Communication, Control and Computing (1999)
19. U.S. Goverment: Standards for privacy of individually identifiable health information, 45 CFR part 160, Part 164 subpart E. In Federal Register, 68(34), pp. 8334-8381 (2003)
20. Baeza-Yates R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison Wesley (1999)