Taxonomy Learning for Semantic Annotation of Web Services

VIORICA R.CHIFU, IOAN SALOMIE, EMIL ŞT. CHIFU
Department of Computer Science
Technical University of Cluj-Napoca
Bariţiu 28, Cluj-Napoca
ROMANIA
http:/utcluj.ro/~cviorica

Abstract: - Web Service composition offers the facility to create new services satisfying a certain functionality which can not be assured by a single existent service. The full potential of Web Service composition can only be achieved if the process is automated. Services should first be annotated with semantic information in order to do automatic service composition. The semantic information is provided by ontologies. This paper presents an unsupervised approach to automatically build a domain specific taxonomy from textual descriptions. The approach is based on hierarchical self-organizing maps. The candidates for concept names are collected by mining text corpora. The term extraction process is based on recognizing linguistic patterns in a text corpus. The taxonomy has been built in the framework of the Food Trace project [20] for traceability in the domain of food industry. The learnt taxonomy represents the domain specific branches of an ontology used for semantic annotation of the web services. Taxonomy concepts are semantic descriptions of the inputs and outputs of the operations provided by a Web Service.

Key-Words: - taxonomy learning, text mining, ontology, semantic web, unsupervised neural network, natural language processing

1 Introduction

A major limitation of the Web services technology is that service discovery and composition still requires manual effort. Yet, the manual composition of Web services is inefficient and impractical on the Internet scale. To motivate this statement we describe a simple Web service application scenario from the meat processing industry: suppose that a Romanian retailer wants to buy a certain quantity of meat products (e.g. salami, sausages) according to the following constraints: the price should not exceed 5 Euro/kg for salami, and 4 Euro/kg for sausages respectively; the delivery should be done within an interval of 5 days; finally, the trasportation price should be of maximum 200 Euro. There is no service to provide such a complex functionality. Instead, there are services that perform the individual functionalities involved. In case the composition is carried out manually, the user has to do the following tasks:

(i) search for producers who are offering the needed products, at the specified price, and with a delivery range of 5 days;

(ii) search for transport companies that assure the trasportation to the desired destination and whose price is not over 200 Euro;

(iii) call the services in the correct order so that the desired result is obtained;

The problem becomes more complex when the user has to specify much more constraints for the compound service or when the number of Web services participant in the composition process increases. This simple example illustrates the need for automatic Web service composition.

The standard language for describing Web services (WSDL) does not provide necessary semantic information for a system to do automatic composition of services. This limitation can be solved by the semantic Web technology, which associates semantic descriptions to each Web service in order to achieve automatic discovery, composition and execution [15]. Semantic Web service descriptions heavily rely on ontologies. An ontology is a "*formal specification of a shared conceptualization*" [24] and provides the necessary concepts for the semantic annotation of Web services.

Ontology building is a time consuming and complex task which requires a high degree of human intervention. This is the reason why nowadays there is a considerable research effort in the domain of automatic ontology building.

This paper presents the automatically building of a domain specific taxonomy out of textual descriptions from Web sites of Romanian meat industry companies. The taxonomy learning is based on hierarchical self-organizing maps [8]. Concept

name candidates are collected by mining text corpora. The term extraction process is based on recognizing linguistic patterns in the text corpus. This taxonomy has been developed in the framework of the Food-Trace project [20] for traceability in the food industry and represents the domain specific branches of an ontology used for semantic annotation of Web services. The concepts in the taxonomy are semantic descriptions of the inputs and outputs of the operations provided by a Web service.

The paper is organized as follows. In section 2 several ontology learning tools similar with the ones used by our approach are reviewed. Section 3 details the implementation of the taxonomy learning tool. Section 4 gives a qualitative evaluation of the experimental results. Conclusions and future directions are presented in section 5.

2 Related work

There is a multitude of ontology learning frameworks [9, 4]. We only enumerate two such frameworks as being the most related to ours. In [1], the terms are represented with distributional (contextual) signatures, similar with our vectors of occurrences in different documents (contexts). The ontology learning is a top-down process, like the behavior of our GHSOM based model. As opposed, the cited work uses decision tree learning, rather than neural learning. A hierarchical self-organizing neural model is used in [12] to arrive at a taxonomy having concept labels only at the leaves. Concept names for the intermediate nodes of the taxonomy are found in a bottom-up process by querying WordNet for common hypernyms of brother nodes.

Our taxonomy learning is based on distributional similarity and clustering [4], where the clustering is neural network driven. Another category of approaches is based on lexico-syntactic patterns, known as Hearst patterns [10], which contain phrases suggesting taxonomic relations: *such as*, (*and* | *or*) *other*, *including*, *especially*, *is a*. In [6, 7] a combination of clustering and Hearst patterns is used.

Most of the clustering based ontology learning approaches use the classical hierarchical clustering algorithm. The neural GHSOM model is better than the classical hierarchical clustering algorithm in terms of speed, noise tolerance and robustness [5], even though any neural model is mathematically more complex.

3 Taxonomy building by using machine learning

The taxonomy contains domain specific concepts which are organized in two taxonomic trees: Product and Feature. The Product tree is a classification of the meat products, and the Feature tree is a classification of the features of these products (such as price, expiration date, storage temperature, quantity). The trees representing our domain taxonomy have been automatically built from a domain text corpus consisting of html pages with information about meat products. The pages were colleted from Web sites of Romanian meat industry companies [18, 19].

The taxonomy learning process has two steps: term extraction, and taxonomy building and pruning. In the *term extraction* step, the relevant terms (words or phrases) for the taxonomy building are extracted from the domain text corpus. These extracted terms become the candidates for the concept names in the final learnt taxonomy.

In the *taxonomy building and pruning* step, the identified terms become concepts, and taxonomic (isA) relations are establish between them, by actually building a tree having the concepts in its nodes. The *pruning* phase avoids the potentially uninteresting concepts for the taxonomy. The flow chart of the taxonomy learning process is illustrated in Figure 1.

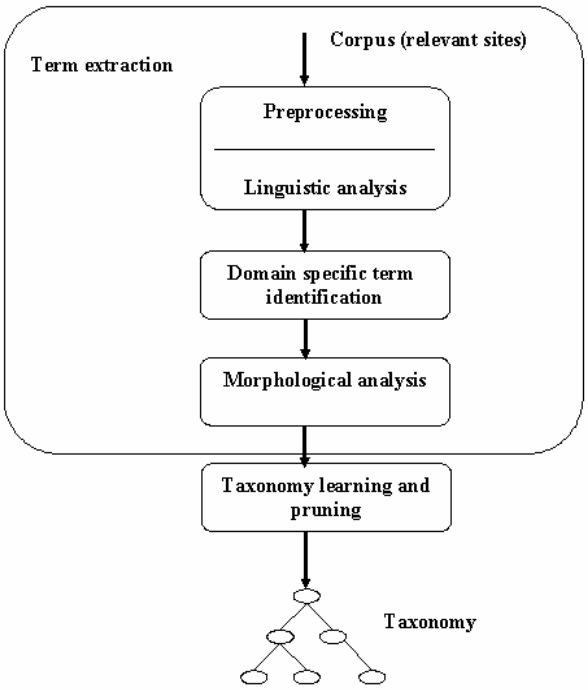

Figure 1: Taxonomy learning process

3.1 Term extraction

The candidates for concept names are identified in a three phase text mining process over the domain corpus. In the first phase a linguistic analysis is performed on the corpus, in the second phase a set of linguistic patterns are applied in order to identify domain specific terms, while in the third phase a morphological analysis is perfomed.

3.1.1 Linguistic analysis

In the linguistic analysis phase, the domain text corpus is first annotated with information about the part of speech (POS) of every word with the help of the Brill POS tagger [3]. Brill tagger is a transformation-based rule tagger that is trainable on different languages. Since the entire ontology, including the domain taxonomy is for the Romanian language, the extracted terms are in Romanian, and the corpus is obviously completely written in the same language.

Brill tagger can only be trained by a supervised learning process starting from an already POS tagged corpus. In order to train Brill tagger for Romanian, we used ROCO, an annotated Romanian text corpus. ROCO contains articles from Romanian newspapers (a collection of 40 million words) collected from the Web over a period of three years (1999-2002). The corpus was tokenized and POS tagged with the RACAI tools [16], having an annotation accuracy of 98%.

ROCO has a tag set of 79 tags for parts of speech and 10 tags for punctuations. Brill tagger trained on ROCO will use part of these tags to annotate our own corpus.

Some corpus preprocessing was required for Brill tagger in order to be able to annotate our corpus [18, 19]. First, we have converted HTML documents to simple text files, by removing all the HTML tags and formatting the text [21]. We have then used a sentence splitter which splitted all the documents in separate sentences displayed one sentence per line [23]. This preprocessed corpus is provided as input to the Brill tagger.

Our original (untagged) corpus consists of 130 documents collected from Web sites of Romanian meat industry companies [18, 19]. Two experiments have been done with the Brill tagger. In the first one, we train the tagger on the whole ROCO corpus. Since the training time was too long, we decided to train the tagger only on part of the articles from the ROCO corpus (13 million words). The evaluation of the trained tagger was performed on our corpus [18, 19]. In this case the accuracy, calculated as the ratio of correct tags out of the total number of the tags, was 81%.

For the second experiment, we split the (untagged) domain corpus into two corpora of equal size. The first one is labeled with part of speech tags after training the Brill tagger with the ROCO corpus. We then used this tagged corpus to train the Brill tagger for use on the second corpus. In this case the accuracy was significantly higher. Table 2 presents the POS tagging results for the second experiment.

The tagging accuracy is lower in the first case due to the lexical ambiguity of the words. The ROCO corpus and our corpus are taken from different domains and some words have different meanings depending on the context in which they appear.

Train corpus	Test corpus	Accuracy
Roco	Maestro	81%
Maestro	CrisTim	92%

Figure 2: Results obtained with Brill tagger

For instance, in the ROCO corpus, the Romanian word “produs” is qualified as *verb*. In our corpus, “produs” is qualified as *noun*. Another example is the Romanian word “exterior”. In the ROCO corpus it is a *noun*, but in our corpus it is an *adjective*.

Below is an example of an output provided by the POS tagger. For the sentence: *“Compozitia: pulpa de porc fara os, sare, condimente naturale”* (Composition: pork pulp without bone, salt, and natural spices), taken from [18], the tagger has identified six nouns and one adjective (AP): *„Compozitia/NN:/COLON pulpa/NN de/C porc/NN fara/S os/NN,/COMMA sare/NN,/COMMA condimente/NNP naturale/AP”*. The first characters of the tag represent the major part of speech (e.g. NN - noun, A - adjective) while the next caracters provide morphological information. For example, NNP is the tag for plural noun and AP the tag for plural adjective.

The corpus annotated in this way is then provided as input to a noun phrase chunker tool to identify domain concepts.

3.1.2 Identifying domain specific terms

The phase of identifying domain specific terms is based on recognizing linguistic patterns (noun phrases) in the domain text corpus. To extract domain specific terms from the corpus, we have implemented a noun phrase (NP) chunker which identifies noun phrases in the linguistically annotated text corpus. The chunker receives as input texts tagged with POS and provides as output tagged texts in which the identified noun phrases are

annotated with a noun phrase tag. Our NP chunker is written by using lex and yacc [22]. A context-free-grammar (CFG) to match noun phrases in Romanian natural language textual descriptions has been defined in yacc. Based on this CFG, a bottom-up parser is generated that uses shift-reduce parsing to recognize the noun phrases.

The written yacc syntax rules of the grammar consist essentially of a head noun together with its pre/post-modifiers (attributes). The pre-modifiers of a head noun can be indefinite determiners and adjectives. The post-modifiers of the head noun can be possessive pronouns, adjectival phrases and prepositional phrases. In the Romanian language, like in the other languages, a noun phrase can be nested within another noun phrase, with no limit on the depth. This nesting process is represented in the grammar by recursive rules. Two kinds of recursive rules can be used to identify such language structures: direct recursive and indirect recursive rules.

We choose to implement our chunker in yacc since we consider that a yacc CFG can capture the most important structural and distributional properties of a natural language (and can also be used to map sentences to abstract representations of meaning).

Our noun phrase chunker works well on the sublanguage of meat processing and product descriptions. For instance, consider the sentence: "*Oferta de produse cuprinde aproximativ 65 de sortimente, punctul forte fiind reprezentat de specialitatile si produsele crud uscate*." (The product offer includes about 65 assortments, the strong point being represented by the specialties and the dry cruel products.) The chunker identifies "*Oferta de produse*", "*sortimente*", "*punctul forte*", "*specialitati*", and "*produse crud uscate*" as noun phrases (see Figure 3).

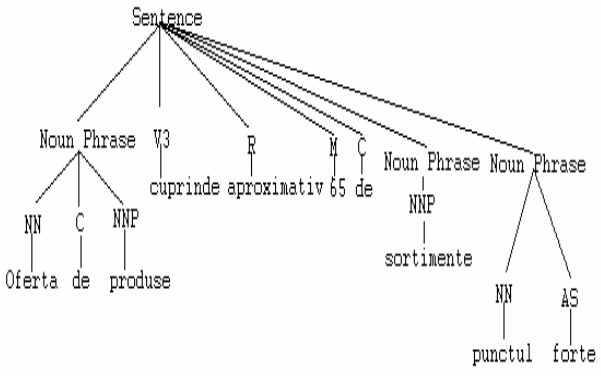

Figure 3: Part of the parse tree of a sentence, with the noun phrases identified by the NP chunker

However, our chunker performance decreases on free Romanian language since natural language is ambiguous (multiple parses can be assigned to one sentence).

3.1.3 Morphological analysis

Romanian language belongs to the Romance language family which also includes Italian, French, Spanish and Portuguese. In the Romanian language, nouns are inflected by gender (feminine, masculine and neuter), number (singular and plural) and case (nominative, accusative, dative, genitive and vocative). Adjectives and pronouns have the same gender, number and case with the noun they modify/refer to. Since the concepts of our taxonomy are designated by noun phrases, we decided to do morphological analysis only for nouns, adjectives and pronouns. By morphological analysis we reduce each inflectional word form to its stem in canonical form.

Our morphological analyzer is written in lex [22]. We have defined regular expressions to identify the various inflectional word forms (nouns, adjectives and pronouns). Starting from the assumption that each word form consists of a stem and its ending (according to the gender, number, case) some replacement rules have been defined for the Romanian language based on the Romanian Language Grammar [11]. For example, one of the rules applies to nouns which end in "uri" and states that the suffix "uri" is removed in the canonical form of the noun. The agglutinative definite determiner for nouns in Romanian with its various inflectional forms is also identified and removed from the suffix of nouns.

The goal of the morphological analysis is to have a unique term originating from its various inflectional forms.

3.2. Taxonomy building and pruning

The taxonomy learning is based on hierarchical self-organizing maps, more specifically, on the Growing Hierarchical Self-Organizing Map (GHSOM) model [8]. In our setting, a learned GHSOM hierarchy is playing the role of a learned taxonomy.

GHSOM is an extension of the Self-Organizing Map (SOM) learning architecture [13] - a popular unsupervised neural network model. The rectangular SOM map is a two-dimensional grid of neurons. Each input data item is classified into one of the neurons in the map. SOM clusters an input data space, giving rise to a similarity based smooth spread of the data items on the map. The data items must be represented as vectors of numerical attribute

values.

The growing hierarchical self-organizing map model consists of a tree-like hierarchy of SOM's [8]. The nodes in the tree are SOM's that can grow horizontally during training by inserting either one more row or one more column of neurons. This happens iteratively until the average data deviation over the neurons in the SOM map decreases under a specified threshold τ_1. The SOM's of the nodes can also grow vertically during training, by giving rise to successor nodes. Each neuron in a SOM map is a candidate for expansion into a successor node. The expansion takes place whenever the data deviation on the current neuron is over a threshold τ_2. The successor SOM map is then trained merely with the data subspace mapped into the parent neuron. The training of the whole GHSOM model converges and stops when both thresholds are satisfied. The depth and the branching factor of the hierarchy learned by GHSOM are controlled by the thresholds τ_1 and τ_2. The GHSOM learning behaves like a top-down process of hierarchical classification of the input data space items.

The noun phrases identified in the corpus are the terms in our setting, and these terms are classified in a GHSOM tree during the process of taxonomy building. To make possible the GHSOM classification of the terms, a vector representation for each term has to be chosen. In our setting, the attributes of the vector representation of a term encode the frequencies of occurrence for the term in different documents of the corpus.

Taxonomy pruning is achieved by avoiding terms occurring in too few documents of the corpus, specifically in less than 1-2% of the total number of documents in the corpus. Such terms cannot be considered as relevant to become concepts of the domain.

4 Experimental results

Below are two of the learned branches of the Product tree (for salami and for ham).

```
{ }
  { produs_fiert_si_afumat_din_piept_de_pui,
    sunca_pui_galinia,  ambalata_in_vid }
  { }
      { crenwurst_extra }  extra wurst
      { produs_crud_uscat_din_carne_de_porc }
      { crenwurst_piept_pui }  chiken chest wurst
      { produs_pasteurizat_din_carne_de_porc }
  { }
      { sunca_praga, sunca }  Praga ham, ham
      { sunca_presata_piept_pui }  chiken chest ham
      { sunca_york, sunca_presata_din_piept_de_pui }
  { produs_pasteurizat_din_carne_porc,
    sunca_presata_toast, sunca_presata }
{ }
```

```
{ salam_turist_extra }  extra tourist salami
{ }
  { salam_chorizo }  Chorizo salami
  { }
      { salam_potcoava }  horseshoe salami
      { salam_de_vara_uscat }  drying summer salami
      { salam_milano, salam_de_porc,
        salam_canadian}  Milano salami, pork salami
      { salam_italian_extra, salam_sicilian,
        salam_piept_pui_galinia, salam_picant_extra,
        salam_taranesc,  salam_sasesc_cu_verdeata,
        salam_sasesc_cu_ceapa, salam_palermo }
{ }
  { salam_rustic, salam_cu_sunca, salam_napoli,
    salam_de_vara_traditional, salam_de_vara_extra }
  { salam_ardelenesc }
  { salam_victoria, salam_sasesc_cu_piper_verde }
{ }
  { salam_sinaia }  Sinaia salami
  { salam }  salami
```

And below is a learned branch for the Feature tree.

```
{ }
  { compozitia, aspectul_exterior,
    tehnologie_de_obtinere, conditie_de_pastrare,
    calitate_organoleptica }
  { termen_de_valabilitate,  expiration date
    recomandare_de_consum} consumption recommendation
  { }
      { condiment_naturale }  natural spices
      { temperatura, umiditate }  temperature,humidity
      { sare }  salt
  { ziua }  day
```

The English translations of the concepts of this taxonomy are given in italics. The concepts – nodes in the taxonomy – are represented as synonym sets, like in a thesaurus. Each node represented by an empty synonym set is a node with no concept label. Such a node can actually be associated with a concept name by querying Romanian WordNet [17] for the most specific concept which is hypernym for all of its immediate successors [6].

5 Conclusions and future work

In this paper we presented an unsupervised taxonomy learning approach to automatically build a domain specific taxonomy from textual descriptions in Web sites of Romanian meat industry companies. The taxonomy has been built in the framework of the Food Trace project [20] for traceability in the domain of food industry. The learnt taxonomy represents the domain specific branches of an ontology used for semantic annotation of Web services. The concepts in the taxonomy are semantic descriptions of the inputs and outputs of the operations provided by a Web service. The experimental results obtained (i.e. the automatically built taxonomy) are encouraging. Different approaches to automatic taxonomy building are hard to evaluate comparatively, since, even if the domain

is the same, authors use different corpora for their experiments. Moreover, our ontology is for the Romanian language, and we can not compare ourselves with other similar approaches from the same domain, because such results have not been reported, yet.

In future work, we plan to extend our ontology learning approach with lexico-syntactic patterns for Romanian (like the English Hearst patterns [10]) and to experiment with other corpora form different domains.

Acknowledgements

This work was supported by the Food Trace project within the framework of the "Research of Excellence" program initiated by the Romanian Ministry of Education and Research.

References:

[1] E. Alfonseca and S. Manandhar, Extending a lexical ontology by a combination of distributional semantics signatures, In A. Gómez-Pérez, V.R. Benjamins, eds., *13th International Conference on Knowledge Engineering and Knowledge Management, LNAI*, Springer, 2002, pp. 1-7.

[2] S. Bechhofer, F. van Harmelen, et al., OWL web ontology language reference, *W3C recommendation*, M. Dean, G. Schreiber, 2004.

[3] E. Brill, A simple rule-based part-of-speech tagger, *ANLP'92, 3rd Confer-ence on Applied Natural Language Processing*, 1999, pp. 152-155.

[4] P. Buitelaar, P. Cimiano, M. Grobelnik, and M. Sintek, Ontology learning from text, *Tutorial at the ECML/PKDD workshop on Knowledge Discovery and Ontologies*, 2005.

[5] G. Chen, S. Jaradat, N. Banerjee, T. Tanaka, M. Ko, and M. Zhang, Evaluation and comparison of clustering algorithms in analyzing ES cell gene expression data, *Statistica Sinica*, Vol. 12**,** 2002, pp. 241–262.

[6] P. Cimiano and S. Staab, Learning concept hierarchies from text with a guided hierarchical clustering algorithm, *ICML workshop on Learning and Extending Lexical Ontologies with Machine Learning Methods*, 2005.

[7] P. Cimiano, A. Pivk, L. Schmidt-Thieme, and S. Staab, Learning taxonomic relations from heterogeneous sources of evidence, In P. Buitelaar, P. Cimiano, B. Magnini, eds. *Ontology Learning from Text: Methods, Applications and Evaluation*, IOS Press, 2005, pp. 59-73.

[8] M. Dittenbach, D. Merkl, and A. Rauber, Organizing and exploring high-dimensional data with the Growing Hierarchical Self-Organizing Map", In L. Wang, et al., eds., *1st International Conference on Fuzzy Systems and Knowledge Discovery*, vol. 2, 2002, pp. 626-630.

[9] A. Gómez-Pérez, and D. Manzano-Mancho, A survey of ontology learning methods and techniques, *OntoWeb Deliverable 1.5*, 2003.

[10] M.A. Hearst, M.A., Automatic acquisition of hyponyms from large text corpora, *14th International Conference on Computational Linguistics*, 1992.

[11] D. Irimia, *Gramatica limbii române*, Polirom Publisher, Bucharest, Romania, 2004.

[12] L. Khan, and F. Luo, Ontology construction for information selection, *14th IEEE International Conference on Tools with Artificial Intelligence*, 2002, pp. 122-127.

[13] T. Kohonen, S. Kaski, et al., Self-organization of a massive document collection, *IEEE Transactions on Neural Networks*, Vol. 11, No. 3, 2000, pp. 574-585.

[14] A. Maedche, S. Staab, Semi-automatic Engineering of Ontologies from Text, *12th International Conference on Software Engineering and Knowledge Engineering*, 2000.

[15] N. Shadbolt, T. Berners-Lee, and W. Hall, The Semantic Web Revisited. *IEEE Intelligent Systems*, Vol. 21, No. 3, 2006, pp. 96-101.

[16] D. Tufiş, Tiered Tagging and Combined Clas-sifiers, In F. Jelinek and E. Nöth (eds.) *Text, Speech and Dialogue, Lecture Notes in Artificial Intelli-gence 1692*, Springer,1999.

[17] D. Tufiş, E. Barbu, et al., The Romanian WordNet, *Romanian Journal on Information Science and Technology*, Dan Tufiş (ed.) Special Issued on BalkaNet, Romanian Academy, Vol. 7, No. 2-3, 2004, pp. 105-122.

[18] http://www.maestro.ro/

[19] http://www.cirstim.ro/

[20] http://www.coned.utcluj.ro/FoodTrace/

[21] http://www.nirsoft.net/utils/htmlastext.html

[22] http://dinosaur.compilertools.net/

[23] http://l2r.cs.uiuc.edu/cogcomp/cc-software.htm

[24] T. Gruber, What is an ontology?, http://www-ksl.stanford.edu/kst/what-isan-ontology.html.