Inductive Logic Programming in an Agent System for Ontological Relation Extraction

M. D. S. Seneviratne and D. N. Ranasinghe

***Abstract*— Ontology plays a vital role in formulating natural language documents to machine readable form on the semantic web. For ontology construction information should be extracted from web documents in the form of entities and relations between them. Identifying syntactic constituents and their dependencies in a sentence, boost the information extraction from natural language text. In this paper we describe the use of Inductive logic Programming as the learning technique used by a multi agent system to perform relation extraction between two identified entities. The learning capability of agents is exploited to train an agent to learn extraction rules from the syntactic structure of natural language sentences. Typed dependencies of the syntactic constituents of sentences provide the background information for the search space to find ingredients for rule induction. In the multi agent system one agent makes use of Inductive Logic Programming for the rule learning process while another agent is expected to use the learnt rules to identify new relations as well as extract instances of predefined relations. All the relations derived are expressed as predicate expressions of two entities. We evaluate our agent system by applying it on number of wikipedia web pages from the domain of birds.**

***Index Terms*—Ontology, Agent, Parser, Annotations, Tagging, Entities, Relations, Predicate, Atom**

I. Introduction

Finding a specific piece of information from a massive collection of web sources is a tedious, time consuming task for a human being. Therefore semantic web researchers have made numerous efforts to make web pages machine readable by annotating the text in web pages with semantic tags and developing ontology to model the information in a more structured manner. Ontology development has emerged as a mean for a standard representation of various types of web pages in the same domain. It is an evolving process and can be extended continuously. Therefore automation or semi-automation of ontology development has become a demanding process.

Ontology describes entities and relations necessary to understand the underlying information. Therefore information extraction for ontology construction mainly involves extracting entities and relations among them, from a natural language text.. Basic information element required for ontology construction is identified as entity. Web is an enormous data repository which provides a rich source of information for domain ontologies. Extracting information from a massive data source is the challenging task in ontology construction/population. Therefore the pioneer task in information extraction for ontology construction is identifying the entities in a natural language document. Information extraction, concept definition from various web sources and text mining are required processes for identifying entities and relations for ontology development. Significant amount of work has been carried out in developing domain specific ontologies. Incorporating ontologies into tools for information access, provide foundation for enhanced, knowledge-based approaches to surveying, indexing and querying of document collections. Many researchers have concentrated on entity extraction. But relation is more complicated and requires heavy linguistic processing. Therefore already established tools in the area are good bases for a commencement of any work towards extracting information for ontology development. In the following section under the Related Work we give a brief description of some of the work carried out recently in this regard.

II. Related Work

A considerable amount of work has been carried out in the area of information extraction at a preliminary stage. Extraction rules generated by various algorithms and techniques are the base for many information extraction systems. Machine learning is the main techniques adopted in information extraction process. Statistical machine learning methods such as Support Vector Machines,[1] Hidden Markov Model [2]etc as well as rule based learning have also been exploited in some research work[3],[23]. Further, work in identifying relations between entities which is more complicated has not yet been progressed satisfactorily. Relation extraction requires heavy linguistic processing of a given text and needs to be addressed in order to complete information extraction process. Many researchers have exploited machine learning,[3],[4],[5],[6],[7],[8],[9] pattern matching,[10],[11],[12],[13] shallow natural language processing [9],[14],[15] and statistical methods[16] in the above mentioned areas. Many systems developed,[17],[18][21] are capable of identifying only taxonomical(is-a) relations. In some systems[18],[22] relation extraction is modeled as categorizing a lexical term into one of the predefined relations.

Two systems Ontosyphon[17] and Text2Onto[18] exploit Hearst phrases template[19] to identify taxonomical relations despite the two different approaches used in achieving the final outcome. Text2Onto develops JAPE(Java Annotation

Manuscript received September 11, 2011, revised September 17, 2011.

M. D. S. Seneviratne is with the Institute of Technology University of Moratuwa, Sri Lanka (e-mail: mdeepika65@gmail.com).

D. N. Ranasinghe is with the University of Colombo School of Computing, Colombo, Sri Lanka (e-mail: dnr@ucsc.cmb.ac.lk).

Pattern Engine)[5] rules within GATE(General Architecture for Text Engineering)[5] whereas Ontosypon analyses sentences to identify the entities wrapped in Hearst phrases. Ontosypon uses an associative learning figure to validate the extracted class instances. Text2onto feeds the identified information to an ontology initiation model to filter out the irrelevant instance occurrences and translates the information in the model to any ontology language. Burcu Yildiz and Silvia Miksch[20] have addressed the issue of adapting their information extraction system in different domains. They have incorporated an ontology management module to tackle different domain ontology to serve this purpose. Their approach uses bag of words and their neighbours, in the rule generation module to generate rules to extract basic concepts based on a predefined/given ontology. Therefore the system can only extract instances for the subclasses and values for the data_type property (i.e. hierarchical relations) in the ontology. OntoMiner[21] uses semantic partitioning to identify taxonomical relations.

Armadillo [22] induces rules for wrappers using irregularities and stores the extracted information in the RDF [23] store as Subject-Verb-Object triplets. Hence the relation extraction is made possible from natural language sentences. Armadillo can easily be switched to different domains. But they have not demonstrated extracting information from complex sentence structure. PubMiner[24] that generates rules based on associative rule discovery technique is capable of extracting both entities and relations from a massive biological literature. Event extractor of PubMiner considers a verb as an event, finds the binary relation between two name entities identified in the sentence where the verb is extracted.. Some systems such as OntoLT[25] and T-rex[26] provide an environment for the user to experiment with various techniques in entity and relation extraction. OntoLT is based heavily on linguistic analysis to identify a head noun and verb to form a predicate expression. T-rex is a test bed for experimenting with extraction algorithms and scenarios.

Snoussi, Magnin and YunNile[27] uses an agent in their tool to extract information from HTML documents and place them in the XML format. A manually constructed definition is integrated into the autonomous agent for the purpose of extracting relevant information. Roxana Danger and Rafeal Berlanga's work[28] concentrates on extracting entity instances from a parsed natural text using OWL[29] ontologies. They use a similarity function between text fragments and lexical description in the ontology to extract entity instances. Several inference rules in the ontology and segment scope definitions that indicates which other segments can be related to a text fragment are applied to add new relations to connect instances. Hoifung Poon and Pedro Domingos[30] propose OntoUSP, a system that learns hierarchical relations over clusters of logical expressions and populates it by translating sentences to logical form. Diana Maynard, Adam Funk and Wim Peters[31] have investigated three linguistic patterns including Hearst patterns for the development of the tool SPRAT in GATE to extract variety of entity types and relations between them.

Both Webkb[32] and the system developed by J.S.Aitken[33] employ Inductive Logic Programming(IPL)[39] system Foil[34] for learning information extraction rules. Webkb uses Foil to learn classification rules to identify entities whereas Atiken uses the same to learn attribute value relations from sentences marked up with relations in the domain. Aitken's ontology based approach is focused on very specific domain; global warming. Background theory to construct rules contains predicates clauses from the text and semantic theory. Success of the system depends on the selected training sentences. Similar algorithm to Foil is used to identify relations defined in the considered ontology by Webkb. R. J Mooney[35] use IPL technique to extract relational patterns. They try to discover rules for Link Discovery which concerns the identification of complex relational patterns that indicate potentially threatening activities in large amounts of relational data. Their approach is completely for a domain specific task.

III. Entity Extraction

For ontology construction, we attempt to extract relations from a text annotated with already identified entities. Therefore it is a must to identify the entities prior to relation extraction.

GATE(General Architecture for Text Engineering) is a framework established for processing texts that provides extensive facilities for researchers in the field. GATE's information extraction tool ANNIE can be used successfully in entity recognition process. Linguistic processing and pattern matching rules are used in GATE for information extraction. ANNIE is bundled with language processing tools Sentence Splitter, Tokenizer and Part of Speech Tagger. Those tools are run on a text to identify the lexical category in which each token belongs, before applying pattern matching rules. The JAPE (Java Annotation Pattern Engine) rules which provide finite state transduction over annotations based on regular expression are used in ANNIE/GATE for entity recognition. The left hand side of a JAPE rule defines regular expressions over which new annotation type is described on the right hand side. GATE framework supports its extensibility by making accommodations for new processing resources added as plug INS.

ANNIE already provides annotations of most general types Person, Location, Job Title etc. We make use of the GATE's developing facilities to build additional plug-ins to the GATE in order to identify domain specific terms representing ontology classes. We cross validate annotated GATE corpus by identifying false entities that enables identification of linguistic features responsible for the extraction of false entities. JAPE rules are then augmented with the counterfactuals of the above mentioned linguistic features to improve the rule accuracy by avoiding false positives. The output of the GATE can be stored outside the GATE framework for further processing when it is embedded in an application. Entire system architecture is given in Fig. 1.

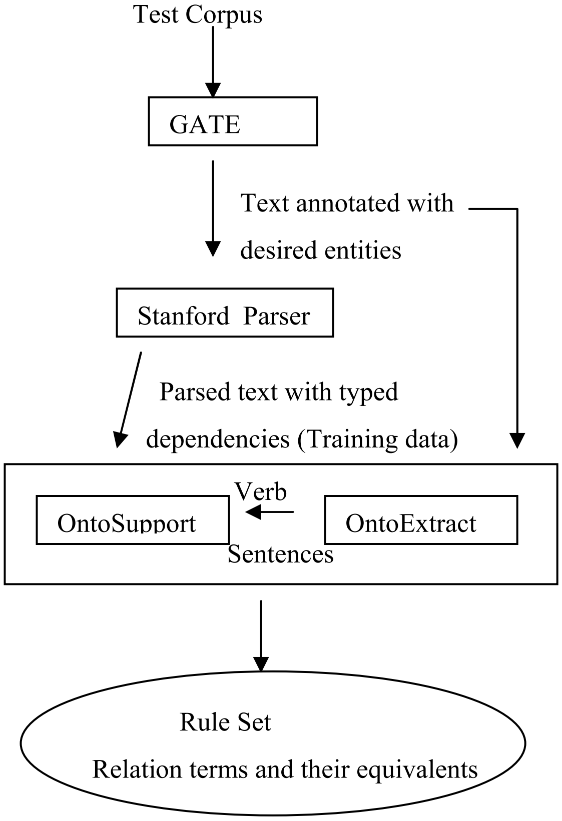
Test Corpus
GATE
Text annotated with desired entities
Stanford Parser
Parsed text with typed dependencies (Training data)
Verb
Sentences
OntoSupport
OntoExtract
Rule Set
Relation terms and their equivalents

Fig. 1 Architecture of the System. Tools used are shown in boxes.

IV. Relation Extraction from Natural Language Text

Successful relation extraction demands heavy linguistic processing. It is not always practical to categorize relationships into few groups because natural language is enriched with a vast vocabulary and a numerous sentence structures. Verb is the powerful lexical term which binds two adjacent syntactic categories and a relation can be defined as a predicate expression of two nouns i.e. subject and object wrapped in syntactic categories as follows.

Verb(Subject, Object) or Verb_Prep(Subject, Object)

Therefore identification of the main verb in a sentence is promising initiative in defining a relation between two entities. For the purpose of relation extraction by verb predicate, documents should be parsed in to identified sentence structures.

For an example the sentence *"Jackdaws are found in Europe, Iran, north-west India and Siberia where they inhabit wooded steppers, woodland, cultivated land pasture, coastal cliffs and villages"* can be mapped to the above predicate format as follows after the sentence is tagged for syntactic constituents and concepts.

located_in(Jackdaw, Europe),
located_in (Jackdaw, Iran),
located_in(Jackdaw, north-westIndia),
located_in(Jackdaw, Siberia)

Inhabit(Jackdaw, woodland),
Inhabit(Jackdaw, wooded steppers)
Inhabit(Jackdaw, cultivated land pastures),
Inhabit(Jackdaw, coastal cliffs),
Inhabit(Jackdaw, villages),

But the complicated nature of the natural language text does not permit to parse the entire text into a set of predefined sentence structures and no human is possibly capable of predefining all the valid syntactic patterns for natural language sentences. Some sentences are very expressive, but contain very little information. Some sentences are short and appear less complicated, but rich in information.

For an example from the sentence which displays the natural language characteristic crossing dependency

"Netball is a ball sport played between two teams of seven players."

We can extract the following 3 relations.

Is_a(Netball, Ball Sport),
Played_between_teams(Netball, 2)
Has_no_of_players(Netball team, 7).

But the above sentence cannot be fitted into a common parse tree. Therefore the system should accommodate uncommon unknown language structures while attempts are being made to fit a sentence to a known structure. In order to accomplice this task the system is required to learn new grammar rules as well as to keep a sentence in place with known grammar rules. But it is very difficult to identify relations accurately from such syntactic structure and grammar rules only.

The Stanford parser[36] that is one of the language parsers available, not only parse a given sentence to give the grammar rules, but give dependencies among linguistic constituents of the sentence also. The Stanford typed dependencies[37] representation was designed to provide a simple description of the grammatical relationships in a sentence that can easily be understood and effectively used by people without linguistic expertise who want to extract textural relations. It represents all sentence relationships uniformly as typed dependency relations in the form of atomic formulas or atoms (i.e. predicate expressions with arguments). These dependencies are quite effective in relation extraction.

For an example for the sentence *"Humming Birds can be found in Cuba including Isle of Youth"* the Stanford parser gives the collapsed dependencies given below.

nn(Birds-2, Humming-1)
nsubjpass(found-5, Birds-2)
aux(found-5, can-3)
auxpass(found-5, be-4)
prep_in(found-5, Cuba-7)
prep_including(Cuba-7, Isle-9)
prep_of(Isle-9, Youth-11)

Highlighted terms in dependencies indicate the already identified entities by the use of GATE and all the terms are syntactically tagged by the parser. From the above structure the relation extracted should be in the form

located_in(Humming Bird, Cuba)
located_in(Humming Bird, Isle of Man)

When generating rules for relation extraction we only have to aim at the dependencies which involve relevant entities identified by GATE. Therefore the typed dependencies are preprocessed to filter the relevant atomic formulas which can contribute to the rule formation. Relevant atoms contains at least one entity instance. When sentences grow in complexity and length the typed dependencies tend to be complicated and vast. Therefore by considering scope of our task some measures are taken in order to reduce the complexity of the typed dependencies of a sentence.

- The atom *“nsubjpass”* is replaced by “*nsubj*”.
- “*prep_including*” is replaced by “*conj_and*” Auxiliary verbs *“aux”* and *“auxpass”* which are non main verbs of the clause such as “be”, “have” etc. are ignored.
- Atoms that represent adjectives, adverbs and determinants are also ignored because there is no significant impact on relations by them.
- If a verb constituent is missing in “*nsubj*” typed dependencies are searched through to find the verb associated with the noun constituent in “*nsubj*”.
- The atom “*det*” is ignored as it indicates the determinants
- Two consecutive nouns contained in the atomic formula “nn” are considered as one noun when one of such noun represents an entity.

For example the reduced typed dependencies of the above mentioned sentence is shown below.

nsubj(found-5, Humming Birds-2)
prep_in(found-5, Cuba-7)
conj_and(Cuba-7, Isle of Youth-9)

Since we use supervised learning to extract relations from the reduced dependencies we use both positive and negative training examples. Semantic ambiguity is one of the difficulties that we come across in natural language processing. For an example main verbs in above mentioned sentences “*found in*” and “*are native*” lead the way to the relation “*located_in*”. Therefore “*are native*” and “*found in*” can be considered as equivalent terms (not synonyms) for “*located_in*” under background information. Therefore the verb constituents from “*nsubj*” of positive examples are added to the set of positive verbs for the relation. Set of negative verbs for the relation is built from the negative examples.

V. IPL Technique Used by Agent System for Relation Extraction

We use the Stanford parser on GATE output which is annotated with the entities, to identify syntactic constituents of a sentence and to derive dependencies among them. These dependencies and syntactic tags provide background knowledge to learn rules for relation extraction. We use an agent OntoSupport to induce rules for relation extraction, searching trough the typed dependencies of natural language sentences given in the training set. Since all the natural language sentences do not fall into predefined solid language structures we cannot provide training examples from all the possible language structures for rule learning process. Therefore learning rules for relation extraction from natural language text is a continuous process. User can expand the training set whenever he finds a different language structure which cannot be covered by the already learnt rules. Autonomous nature of the agent technology permits the agent to update the rule base and the knowledge while running in the background when the user updates the training set.

We use another agent OntoExtract to extract information for ontology construction by applying the rules formed by OntoSupport. When OntoExtract is released on the internet it can not only extract information for different users but can provide OntoSupport some information also in order to update its knowledge and rule set. We use JADE[38]; an agent framework to implement our agents.

A. Learning Extraction Rules by Agent OntoSupport

The agent OntoSupport learn rules to extract relation instances for a known relation such as *located_in, part_of, feed_on* etc, some of which are domain specific relations. The outcome of the Stanford parser is used by OntoSupport in order to derive rules for relation extraction. OntoSupport employs IPL technique to derive the set of rules based on the text annotated with the entities.

Since Stanford parser provides many atomic formulas or atoms in the form of typed dependencies as well as syntactic tagging the output of the Stanford parser is a good candidate for inductive logic programming. In inductive logic programming the rules are induced with the available atoms and are generalized with respect to positive training data. Rules are specialized with respect to negative training data. We have a set of positive and negative training examples along with syntactic constituents (syntactic tags) of the sentence from which the relation is extracted and a set of atoms in typed dependencies.

For examples the sentence *Humming Birds can be found in Cuba including Isle of Youth* gives a positive instance for the relation located_in resulting located_in(Humming Bird,Cuba) and located_in(Humming Bird, Isle of Youth) . The sentence *Cranes live on all continents except Antarctica and South America* is an evidence for negative relation instances and the extraction can be represented as ¬located_in(Cranes, Antarctica) and ¬located_in(Cranes, South America).

Therefore we can define the set of positive training data E^+, the set of negative training data E^- and the background information B for ILP.

E^+ = {Positive relation instance pairs for the relation}
E^- = {Negative relation instance pairs for the relation}
B = Reduced Stanford parser output , Syntactic tags

We adopt our ILP algorithm from the attribute value learning system which uses NEWGEM propositional learner[39], to induce rules from the available atoms given by Stanford parser in order to cover all the positive training data. Relation is represented by the left hand side of the rule and the body of the rule is by the right hand side of the rule as an attribute value expression. Body of the rule contains one or more conditions for the relation to be fulfilled when applied on reduced Stanford typed dependencies. Since typed dependencies are already in the predicate form there is no requirement of transforming them into a suitable form for the application of ILP. Rules are specialized with respect to negative examples by adding atoms from typed dependencies of negative data if the rules can extract any of the negative relations. In deriving the preliminary set of rules we take the atom relevant to subject noun *nsubj* out from the set of available atoms as it is present in almost all the sentences. Initial rule set is formed with two atoms; *nsubj* and an another atomic formula from the dependencies. Then a rule is in the following format.

Relation(Entity1, Entity2) :- nsubj(verb, Entity1),
(atom(….,Entity2.) ∨
atom(Entity2,…..))

Separate rules are constructed at the disjunction between conditions. Rule body allows only internal disjunction (i.e. disjunction between attribute values) appeared in a rule.

TABLE 1 shows attributes and values of two positive example and one negative example from the relation located_in(bird,location) where bird and location are entity classes. For relation examples we consider a relation as a class for ILP method.

TABLE I: ATTRIBUTES AND VALUES FOR THE RELATION LOCATED_IN

Class	nsubj(x,y)		conj_and (x,y)		prep_in (x,y)		Prep_except (x,y)	
	x	Y	x	y	x	Y	X	y
Located_in (bd,lo)	verb	bd	lo	lo	verb lo	lo lo		
Located_in (bd,lo)	verb	bd			verb verb	lo lo		
¬located_in (bd,lo)	verb	bd			verb verb	noun lo	verb	lo

bd - bird, lo – location

The representation of typed dependencies is already in the predicate form and can easily be converted into a way suitable for attribute value learning. The attributes and the values from the propositional form for the target relation located_in(bird, location) are shown in TABLE II. It shows the instances of entity classes bird and location for two positive examples and one t negative example and the dependencies relevant to each example. Entities can be considered as attributes and instances of entities give the values. The variables x, y denote the entity class instances while the variable z indicates the syntactic category of another term which associates with entity instances. The presence of entities in dependency clauses is the major factor in identifying a relation and the lexical terms represented by z does not play a significant role in rule generation.

The rule generation according to our algorithm is not initiated by a seed (a positive example) as in NEWGEM. Instead it collects elements (i.e. atomic formulas from typed dependencies) for the rule formation from the background information of the set of positive examples and places them in a list. We use a heuristic approach to create the list in finding the atomic formula to combine with *nsubj* to form rules. Most occurring atoms are given the priority to join with *nsubj* to form a rule and the most generalized rule is formed first. Therefore the first task of the OntoSupport is to create the list of atoms and sort the list according to the number of times that an atom occurs. The beam is the sorted list of atoms and not exactly the body of a rule. Rule bodies are formed using beamsearch algorithm, taking atoms from the beam to combine with *nsubj* and specialized with respective to negative training data. Therefore in finding the best body the best atom from the beam is taken to form the rule body.

TABLE II: PROPOSITIONAL REPRESENTATION

Class	Variables			Propositional Features					
	Bd	Lo	Oth-er z	Nsubj (z, bd)	Conj_and (z,lo)	Conj_and (lo, Lo)	Prep_in (z,lo)	Prep_in (lo, Lo)	Prep_except (z,lo)
+	Humming bird	Cuba, Isle_of_youth	verb	true	false	false	true	true	false
+	Parrot	America, Australasia	verb	true	false	true	true	false	false
-	Pooto	Chile	verb noun	true	false true	false	true	false	true

From the positive examples shown above in the table the initial set of rules can be formed as follows.

located_in(Humming Bird, Cuba):- nsubj(verb, Humming bird),
prep_in(verb,Cuba)

located_in(Humming Bird, Isle_of_Man):- nsubj(verb, Humming Bird),
prep_in(verb, Isle_of_Man)

located_in(Parrot, America) :- nsubj(verb, Parrots),
prep_in(verb, America),
conj_and(America, Australasia)

The number of rules can be reduced by generalizing rules. In generalizing some rules become redundant and some rules can be joined with another rule by internal disjunction.

The following shows the generalized form of the above mentioned rules

Located_in(Bird,Location) :- nsubj(z,Bird,
prep_in(z,location)
Located_in(Bird,location) :- nsubj(z,Bird),
conj_and(location,location)

Both 1st and 3rd rules cover the negative example. Therefore those rules are augmented with negation of the clause specific to negative example to uncover the negative example but to still cover the all positive examples. Then the 1st and 3rd rule will be modified as follows.

```
Located_in(Bird,location) :-  nsubj(z,Bird),
                              prep_in(z,location),
                              ¬prep_except(z,location)

Located_in(Bird,location):- nsubj(z,Bird),
                            conj_and( location,location),
                            ¬prep_except(z,location)
```

Algorithm for ILP

Covering Algorithm

```
Create a list of atoms ordered according to the number of occurrences
(i.e. most occurred atom at the head and least occurred atom at the end)
Initialize the LHS of the rule   LHS = Relation
Repeat
  Call the BeamSearch algorithm to find the best body BestBody
  For all training data in $E^+$
      Apply the BestBody
      Remove the covered positive examples from $E^+$.
    Add the BestBody to the rule set
Until .$E^+ = \theta$
```

Since the maximal general rules are formed initially due to the language bias[39] in the rule learning criteria the initial rule body is specialized conveniently by extending it with other atomic formulas. It is augmented with counterfactuals of the atoms specific to covered negative examples to cover the positive examples but not to cover any negative examples. If there are no such atoms to be found the main verb of the sentence is considered as a negative verb and the set of the negative verbs is updated with the found verb.

For e.g. the sentence *Cranes live on all continents except Antarctica and South America* which indicates negative relations ¬located_in(Crane Antactica) and ¬located_in(Crane South America) gives reduced typed dependency

nsubj(live-2, Cranes-1)
prep_on(live-2, continents-5)
conj_and(Antarctica-7, South America-9)
prep_except(continents-5, South America-10)

From the above dependency prep_except can be considered as the specific atom for ¬located_in and it normally does not occur for located_in.

The sentence *Swans are absent from tropical Asia, Central America, northern South America and the entirety of Africa* gets the following dependencies from the Stanford parser.

nsubj(absent-3, Swans-1)
prep_from(absent-3, tropical Asia-6)
prep_from(absent-3, Central America-9)
conj_and(tropical Asia-6, Central America-9)
amod(America-13, northern-11)
prep_from(absent-3, America-13)
conj_and(tropical Asia-6, America-13)
prep_from(absent-3, entirety-16)
conj_and(tropical Asia-6, entirety-16)
prep_of(entirety-16, Africa-18)

In this example there are no atoms specific to the covered examples. The verb “absent” will be added to the set of negative verbs for the relation.

BeamSearch Algorithm

```
Intialize RHS to Head_of_List
RHS = RHS & Head_of_Tail
Remove Head_of_Tail from the list
For all training data in $E^-$
    Apply RHS
    If a negative example is covered
      add the complement of an atom speacialized to the
      negative example to the RHS to uncover the negative
      example but cover the positive examples.
    If there are no such atoms
       add the verb to the set of negative verbs,
    add the clause ¬negative(verb) to the RHS.
 BestBody = RHS
```

Since the beam contains the elements for rule construction, the size of the beam is not a significant factor for the efficiency of our method as in NEWGEM algorithm. The measure of rule quality determines the partial order of the rules and we define the parameter Lexical Evaluation Function (LEF) to assign priorities to rules. The measure consists of a list of criteria that are applied to each rule. When the first criteria cannot discriminate between two rules, the second one is used and so on. We use the following list of criteria that order rules from the best to worst.

- higher number of covered positive examples
- lower number of conditions in the rule
- earlier constructed rule
- rule contained a negation of an atomic formula

When both positive and negative training data sets are updated with new additions, OntoSupport checks for the compatibility of existing rules with the new data. In this type of learning the agent remembers all the training examples seen so far, as well as the rules it formed. New rules are guaranteed to be consistent and complete with respect to all training data. Information regarding any unknown sentence structure of known entities is conveyed to OntoSupport by OntoExtract. Then OntoSupport updates the rule set positive and negative verb sets. The positive and negative verb sets can be used as a secondary measure when OntoExtract finds that a sentence of two known entities cannot be covered by the extraction rules of a particular relation.

A. Relation Extraction by OntoExtract

The task of the agent OntoExtract is to apply the rules generated by OntoSupport to extract relations in a given corpus of a particular domain. In addition OntoExtract has the ability to process the sentences of entities not extracted as a relation in order to find whether the entities form negative relation or a new relation. Such a sentence can be categorized into one of the followings.

(i) Verb unknown but extraction rules cover the typed dependencies

(ii) Verb known but extraction rules cannot cover the typed dependencies.

(iii) Verb unknown and extraction rules cannot cover the typed dependencies.

From the sentences fallen into group (i) OntoExtract communicates the verb constituent in the "*nsubj*" to OntoSupport that can update the set of positive verbs for the relation with the verb sent. Sentences in the category (ii) give a different structure for the relation. Then OntoExtract sends the URL of the file where the sentence and its syntactic constituents are stored, to OntoSupport to form a rule to cover newly found sentence structure for the relation. Sentences in the category (iii) form a completely new relation and they are sent to OntoSupport to formulate the new relation.

VI. RESULTS

We have used the domain of birds to test our system. Creation of ontology for the domain of birds requires to establish domain specific entities and relations between them. We identify entities *Bird, Location, Body_part, Colour, Diet, Habitat, Size, No_of_eggs, Characteristic* etc and attempt to find relations existing between them.

First we have selected rather small set of training data which cover different complicated sentence structures (13 wikipedia web pages as training data and 14 wikipedia web pages as testing data). From the training data the OntoSupport learnt the rules shown in fig. 2 for the relation *located_in()* which exists between Bird and Location. While the agent is in action it is expected to learn more rules in the case of any deviation from the already created rules.

TABLE III and TABLE IV show the positive and negative examples respectively for the relation *located_in(Bird, Location)* in our training set which is taken from 13 Wikipedia documents.

TABLE III: POSITIVE TRAINING DATA

Bird	Country
Ostriches	Africa
Humming Birds	Cuba
Humming Birds	Isle of Man
Parrots	South America
Parrots	Australasia
Doves	Indomalaya
Doves	Australasia
Emu	Australia
Eagles	California
Shoebill	Africa
Jackdaws	Iran
Jackdaws	India
Jackdaws	Siberia
Nutcracker	Europe
Nutcracker	Asia
Potoos	Mexico
Kiwi	New Zealand

TABLE IV NEGATIVE TRAINING DATA

Bird	Country
Cranes	Antarctica
Cranes	South America
Ostriches	Middle East
Swans	Asia
Swans	Central America
Swans	South America
Swans	Africa
Potoos	Chile

The set of rules in the Fig. 2 is generalized to reduce the number of rules. The final set of rules is given in Fig.3

```
located_in(Bird, Country): -nsubj(VB,Bird)
                            conj_and(Locationy, Locationy),
                           ¬prep_except(NN, Location),
                          ¬negative(VB)

located_in(Bird, Country) :- nsubj(VB, Bird),
                             conj_and(VB, Location),
                            ¬prep_except(NN, Location),
                            ¬negative(VB)

located_in(Bird, Location) :- nsubj(VB, Bird),
                              conj_and(Location, NN),
                             ¬prep_except(NN, Location)
                           , ¬negative(VB)

located_in(Bird, Country) :- nsubj(VB, Bird),
                             conj_and(NN, Location),
                            ¬prep_except(NN, Location),
                            ¬negative(VB)

located_in(Bird, Location) :- nsubj(VB, Bird),
                              prep_in(VB, Location),
                             ¬negative(VB)

located_in(Bird, Location) :- nsubj(VB, Bird),
                              prep_to(VB, Location),
                             ¬negative(VB)

located_in(Bird, Location) :- nsubj(VB, Bird),
                              prep_to(NN, Location),
                             ¬negative(VB)

located_in(Bird, Location) :- nsubj(VB, Bird),
                              prep_from(VB, Location),
                             ¬negative(VB)
```

Fig. 2

OntoExtract applied the rules on 14 text documents and found relations shown in TABLE V.

```
located_in(Bird, Country) :- nsubj(VB, Bird),
                              conj_and(X, Location),
                             ¬prep_except(Y, Location),
                              ¬negative(VB)

located_in(Bird, Location) :- nsubj(VB, Bird),
                              conj_and(Location, X),
                             ¬prep_except(Y, Location),
                             ¬negative(VB)

located_in(Bird, Location) :- nsubj(VB, Bird),
                              prep_in(VB, Location),
                             ¬negative(VB)

located_in(Bird, Location) :- nsubj(VB, Bird),
                              prep_to(X, Location),
                              ¬negative(VB)

located_in(Bird, Location) :- nsubj(VB, Bird),
                              prep_from(X, Location),
                              ¬negative(VB)
```

Fig. 3

TABLE V RELATIONS EXTRACTED BY ONTOEXTRACT

Relation Instances found for the relation located_in()	Negative relation Instances for the relation located_in	New Relations established between bird and location
(Albatross, Southern Ocean) (Petrel, Southern Ocean) (Eagle, Eurasia) (Flamingo, America) (Nutcracker, Europe) (Nutcracker, Asia) (Macaw, Mexico) (Macaw, Caribbean) (Hornbill, Africa) (Hornbill, Asia) (Junglefowl, Sri Lanka) (Junglefowl, India) (Cassowary, New Guinea) (Kakapo, New Zealand)	(Pelican, Antarctica) (Pelican South Pacific) (Cuckoo, South America) (Cuckoo, Middle East) (Cuckoo, North Africa) (Owl, Antarctica) (Woodpecker, Australasia) (Woodpecker, Madagascar) (Woodpecker, Antarctica)	farmed_in is_dangerous is_national_bird

VII. CONCLUSION

In this paper we have discussed the use of linguistic characteristics combined with the Inductive Logic Programming Technique to learn rules for relation extraction. IPL plays the major role as the learning algorithm of the agent OntoSupport to induce relation extraction rules. A set of rules for relation extraction is learnt from the typed dependencies of training sentences (i.e. annotated text with entities and relations). From the semantic annotations on the sentence the agent identifies various equivalent terms for a relation and continuously updates its knowledge throughout the operation in order to reduce the effects of semantic ambiguity. Inductive logic programming used in the agent's learning prevents the agent extracting negative relations. Both taxonomic and non taxonomic relations are treated in the same way as typed dependencies is a very reliable source for finding predicate clauses specific to different relations. The biasness of our data set, selecting only from Wikipedia does not have an adverse impact on the final outcome as it is a good hierarchical information source for many domains. A rather small test corpus which covers a number of different syntactic structures is used for training at the beginning. But when agents are in action the system continuously learns new rules and updates the knowledge wherever appropriate. Another positive aspect of our approach is that the relations which cannot be categorized in to pre defined relations can specifically be identified. Therefore there are no relations of unknown category. The same set of rules can be tried in different domains as well as the same techniques are applicable to different domains. Then the entity types will be replaced with the entities specific to a domain if the rules comply with any of the annotated sentence. In this project we manage to exploit linguistic characteristic to be used by IPL within the agent technology for successful relation extraction.

ACKNOWLEDGMENT

Sincere thanks goes to Prof. A. Karunananda of Faculty of Information Technology, University of Moratuwa for his valuable contribution.

REFERENCES

[1] J. S. Taylor, N Cristianini, "Support vector machine and other kernel-based learning methods," Cambridge University Press, 2000.

[2] D. Ramage, "Hidden markov models fundamentals," Available: http://www.stanford.edu/class/cs229/section/cs229-hmm.pdf

[3] F. Ciravenga., "(LP)^2, An adaptive algorithm for information extraction from web-related texts," Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management,2001.

[4] A.C.Knoblock, K.Lerman, S. Minton, I Muslea, "A machine learning approach to accurately and reliably extracting data from the web," IJCAI-2001 Workshop on Text Learning: Beyond Supervision, Seattle, 2001.

[5] H. Cunningham., D. Maynard., K..Bontcheva,. and V.Tablan , "GATE: An architecture for development of robust HLT applications.," In Proc. of the 40th Anniversary Meeting of the Association for Computational Linguistics, pp 168-175, 2002

[6] C. A. Knoblock, S. Minton, J. L. Ambite, N. Ashish, P.J. Modi, I. Muslea, A. G. Philpot, S. Tejada, "Modeling web sources for information integration," Proc. Fifteenth National Conference on Artificial Intelligence, 1998.

[7] H. Han, C.L. Giles, E. Manavoglu, H. Zha, "Automatic document matadata extraction using support vector machines," Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries, pp 37-48, Houston, Texas, 2003.

[8] N. Kiyavitskaya, N. Zeni., R.James., L.Mich., J..Mylopoulos., "Semi-automatic semantic annotations for web documents," Proceedings of "SWAP 2005", 2005.

[9] M.D.S. Seneviratne, D.N. Ranasinghe, "Use of agent technology in relation extraction for ontology construction," Proceedings of 2011 4th IEEE International Conference on Computer Science and Information Technology, Vol.4, June 2011, pp 70-76, Chengdu, China.

[10] F. Ciravenga. and Y. Wills, " Designing adaptive information extraction for the semantic web in Amilcare, annotation for the semantic web," in the Series Frontiers in Artificial Intelligence and Applications by IOS Press, Amsterdam, 2003.

[11] M. Dzbor, J. Domingue, E. Motta, "Magpie-toward a semantic web browser," Proc. of the International Semantic Web Conference, pp 690-705, 2003.

[12] B. Papov, A. Kiryakov, D. Manov, A. Kirilov, D. Ogniyanoff, M. Goranav, " Towards semantic web information extraction" Available:

http://gate.ac.uk/conferences/iswc2003/proceedings/popov.pdf

[13] A. Wasilevska "Apriori Algotithm" Available: http://www.cs.sunysb.edu/~cse634/lecture_notes/07apriori.pdf

[14] P. Buitelaar., D. Olejnik. and M.Sintek., "OntoLT: A protégé plug-in for ontology extraction from text.," Proceedings of the International Semantic Web Conference, pp 31-44, 2003.

[15] D. Celjuska., M. Vargas-Vera., "Ontosophie A Semi-automatic system for ontology population," from text, International Conference on Natural Language Processing, 2004. Available: http://kmi.open.ac.uk/publications/pdf/kmi-04-19.pdf

[16] S. Handchuth, S. Staab, F. Ciravenga., S-CREAM – Semi-automatic CREAtion of matadata," The 13th International Conference on Knowledge Engineering and Management, pp 358-372, 2002.

[17] L.K.Dowell, M.J.Cafarella., "Ontology-driven information extraction with OntoSyphon," International Semantic Web Conference, 2006.

[18] P. Cimiano., J. Volker., "Text2onto – a framework for ontology learning and data driven change discovery," Int. Conf. on Applications of Natural Language to Information Systems, 2005.

[19] M. Hearst., "Automatic acquisition of hyponyms from large text corpora, in proceedings of the 14th International conference on Computational Linguistics, 1992.

[20] B. Yildiz., S. Miksch, "Motivating ontology-driven information extraction," Proceedings of the International Conference on Semantic Web and Digital Libraries (ICSD-2007), 2007.

[21] H. Davalcu.,S. Vadrevu, S. Nagarajan.,"OntoMiner: Bootstrapping and populating ontologies from domain specific web sites," IEEE Intelligent Systems 18(5), pp 24-33., 2003.

[22] F. Ciravenga., S. Chapman, A. Dingili., Y. Wilks., (2004) "Learning to harvest information for the semantic web," Proceedings of the 1st European Semantic Web Symposium, pp 312-326, Greece, 2004.

[23] RDF Vocabulary Description Language Available: http://www.w3.org/TR/rdf-schema/

[24] J. H. Ecom., B.T. Zhang, "PubMiner: Machine learning-based text mining for bio medical information analysis," Artificial Intelligence: Methodology, Systems, Applications, 2004.

[25] P. Buitelaar, D. Olejnik and M. Sintek, "OntoLT: A protégé plug-in for ontology extraction from text.," Proceedings of the International Semantic Web Conference 2003.

[26] J. Iria and F. Ciravenga, "Relation extraction for mining the semantic web," Proceedings Machine Learning for the Semantic Web, Dagstuhl Seminar 05071, Dagstuhl, DE, 2005.

[27] H.Snoussi, L. Magnin, and J.Y. Nie, "Toward an ontology- based web data extraction," The AI-2002 Workshop on Business Agents and the Semantic Web, 2002.

[28] R. Danger and R. Berlanga, "Generating Complex Ontology Instances from Documents," Journal of Algorithms, vol. 64, Issue 1, Jan. 2009, pp 16-30.

[29] OWL Web Ontology Language Semantics and Abstract Syntax Available: http://www.w3.org/TR/owl-features/

[30] H. Poon, and P. Domingos, "Unsupervised Ontology Induction from Text," ACL'10 Proc. of the 48th Annual Meeting of the Association for Computational Linguistics, pp 296-305, 2010

[31] D. Maynard, A. Funk and W. Peters, "SPART: a tool for automatic semantic pattern- based ontology population," Proc. of International Conference for Digital Libraries and the Semantic Web, 2009

[32] M. Craven, D. DiPasquo, D. Freitag, A.K. McCallum, T. M. Mitchell, K. Nigam, S. Slattery, "Learning to construct knowledge bases from the world wide web," Artificial Intelligence 118(1/2) 69-113, 2000.

[33] J.S. Aitken, "Learning information extraction rules: an inductive logic programming approach," Prpceedings of 15th European Conference on Artificial Intelligence, pp 355-359, 2002, Lyon, France

[34] J.R. Quinlan, R. M. Cameron-Jones, "FOIL: A midterm report," In Proceedings of the European Conference on Machine Learning, pp 3-20, Vienna, Austria, 1993.

[35] R.J. Mooney, P. Melville, L.P. Tang, J. Shavlik, I.D.C. Dutra, D. Page, V.S. Costa, "Relational data mining with inductive logic programming for link discovery," Proceeding of the National Science Foundation Workshopon Next Generation Data Mining, Nov. 2002, Baltimore, MD.

[36] http:/nlp stanford edu/software/lex-parser.shtml.

[37] M.C.D. De Maneffe. and C.D. Manning., "Stanford Typed Dependancies," Manual, 2008.

[38] Jade Java Agent Development Framework Available: http://jade.tilab.com/

[39] N. Lavrac., and S. Dzeroski , "Inductive Logic Programming: Techniques and Applications," Ellis Horwood, New York, 1994.

M.D.S. Seneviratne graduated with B.Sc.(Eng) in the field of materials engineering from University of Moratuwa Sri Lanka in 1991 and obtained her Masters degree in computer science from University of Wales College of Cardiff in 1996.

She worked as an instructor at the Department of Materials Engineering University of Moratuwa before leaving to United Kingdom. After returning from UK she joined Informatics Institute of Computer Technology which was affiliated to Manchester Metropolitan University, as a lecturer. At present she works as a lecturer at the Institute of Technology University of Moratuwa and conducts research for her M.Phil. degree. She also worked as a visiting lecturer for University of Sri Jayawardhanapura Sri Lanka and Royal Institute which awards degrees in collaboration with University of London. She has published a project report on the title "Use of paddy husk in insulation bricks" as fulfillment of her first degree and the thesis of the research carried out on "Novel Applications of Image Processing" as a part of her masters degree. The research paper titled "Use of Agent Technology in Relation Extraction for Ontology Construction" that she has written with Dr. D. N. Ranasinghe appears in the Proceedings of 2011 4th International Conference on Computer Science and Information Technology.