2014 International Conference on Advances in ICT for Emerging Regions (ICTer): 142 - 148

Natural Language Dependencies For Ontological Relation Extraction

M.D.S.Seneviratne[1], D.N.Ranasinghe[2]

[1]*Institute of Technology University of Moratuwa Katudedda, Moratuwa, Sri Lanka*

[2]*University of Colombo School of Computing. Reid Avenue, Colombo 07, Sri Lanka*

[1]mdeepika65@gmail.com

[2]dnr@ucsc.ac.lk

Abstract— **Natural Language Processing techniques play an essential role in extraction of necessary information for ontology construction from unstructured text. Identifying syntactic constituents and their dependencies in a sentence, boost the information extraction from natural language text. Main ingredients required for ontology construction are required to be extracted from text in the form of entities and relations between them. Language dependency constructs that express the binary dependencies of the lexical terms in a sentence can be considered as good indicators in identifying binary relationships existing between entities. In this paper we describe that how the typed dependencies produced by a natural language parser are used by a multi agent system to generate rules for relation extraction between two identified entities. The typed dependencies produced by parsing are processed to eliminate unnecessary dependencies and make them more appropriate to be used in rule learning for relation extraction. 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.**

Keywords— **Ontology, Agent, Parser, Annotation, Tagging, Typed Dependencies, Entities, Relations**

I. INTRODUCTION

The web being the latest and fastest information provider, people prefer accessing the web for their information needs. Finding a specific piece of information from a massive collection of unstructured web sources is a tedious, time consuming task for a human being. Therefore the need of Semantic Web which provides a queriable information and knowledge layer by attaching meanings to information, has become vital. Ontology is a strong concept that bridges the gap between the semantic web and unstructured natural text. The machine can access the ontology and provide information required by users saving the user from laborious task of searching numerous web sources and surfing through the jumble of natural text to find a piece of information.

Ontology development requires identifying mainly entity classes, class instances and relations between entities wrapped in natural language texts. Since the natural text is vast in terminology and sentence patterns, extracting information wrapped in natural language sentences in order to model them into the structured format of ontology is a complex and continuous process. As such the analysis of natural language sentences syntactically and semantically plays an important role and can become necessary preprocessing step in successful information extraction. Parsing has become an important first step in natural language processing and it categorizes lexical terms into syntactic constituents. Semantic ambiguity in natural language is added to the complexity of language processing and the processing techniques cannot expect a success without addressing those issues. Therefore semantic web researchers have made numerous efforts to develop methodologies to process natural language texts towards ontological information extraction. Significant amount of work has being carried out in entity extraction; but relation extraction is yet to be addressed extensively.

II. RELATED WORK

Many researchers have exploited machine learning [1],[2],[3],[4],[5], pattern matching[6],[7],[8],[9], shallow natural language processing [10],[11] and statistical methods[14] in the area of information extraction. Relation extraction requires heavy linguistic processing of a given text and needs to be addressed in order to complete information extraction process. Therefore many systems have used shallow natural language processing in combination with other methods. Few systems have exploited language dependencies to develop methodologies to identify relations existing between known entities.

Two systems Ontosyphon[13] and Text2Onto[14] exploit Hearst phrases template[15] to identify taxonomical relations despite the two different approaches used in achieving the final outcome. Text2Onto develops JAPE(Java Annotation Pattern Engine)[16] rules within GATE(General Architecture for Text Engineering)[18] to cover Hearst phrases 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.

Armadillo[17] induces rules for wrappers using irregularities in the text and stores the extracted information in the RDF[18] 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[19] 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[20] and T-rex[21] 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

978-1-4799-7732-1/14/$31.00 ©2014 IEEE

expression. T-rex is a test bed for experimenting with extraction algorithms and scenarios. Diana Maynard, Adam Funk and Wim Peters[22] 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.

T. Wang @ al[23] have addressed hierarchical relation extraction using Support Vector Machine (SVM)[24] based approach. They have experimented on ACE2004[25] training data which defines a hierarchy of relations with 7 top types and 22 sub types. SVM models are created by feature vectors generated during the training phase with comparatively large number of features derived from number of sources including GATE-based language processing tools. The linear kernel has been used with SVM model to extract relations. Aron Culotta and Jeffrey Sorensen[26] use dependency tree kernel within SVM and they also use ACE training corpus of news articles to experiment with. They use the smallest common sub tree in dependency tree that includes both entities and define a set of features for each node to be used in the feature vector for the application of tree kernel. A similar approach is taken by R.C. Bunescu and R.J. Mooney[27] by using the shortest path between the entity instances in the dependency tree to build a kernel function to apply with SVM. All the three SVM based systems which we describe here concentrates on extracting instances for pre defined relation types as defined in ACE guidelines. Therefore there are hardly any provisions to identify a relation fallen outside predefined types. Culotta and Sorensen's tree kernel method makes the system more flexible by using a sparse tree kernel allowing non matching nodes within matching subsequences at the cost of computational time. Although Bunescu and Mooney's shortest path kernal shows an improvement over tree kernal both paths in the comparison should have same length according to their kernel function.

The method RelEx[28] developed by K. Fundel @ al, medical abstracts uses simple rules to identify relations connecting pairs of proteins from dependency parse trees and it's application is restricted to medical abstracts. Wanderlust[29] is a system developed to extract semantic relations from natural language text using dependency grammar patterns. Wanderlust system generates linguistic patterns for linked paths between entity instances, from the language dependencies and the patterns are used to extract relations from Wikipedia documents. Characteristics of the Wikipedia presentations are exploited by Wanderlust in order to enhance the extraction mechanism. Wanderlust labels the relations in the predicate form combining one or more lexicons as identified from the link paths. Therefore there is a tendency of extracting unhelpful nonsensical relations in addition to false relations and same relation type can get named by different labels.

None of the systems that we have discussed here, have made use of the typed dependencies (i.e. dependency label between the governor and the dependent) of the lexical terms in a sentence. Certain typed dependencies such as determiners, composite noun constructs etc do not account for a relation. Therefore we can eliminate unnecessary dependencies when developing techniques for relation extraction especially with non local dependencies (long distance dependencies). In our work we intended to exploit the typed dependencies which can be readily obtained on the application of Stanford parser[30], in order to construct methodologies for relation extraction for ontology development.

III. EXISTENCE OF RELATION VERBS IN NATURAL LANGUAGE TEXT

Identification of the main verb in a sentence is a promising initiative in defining a relation between two entities. 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)

But extraction of verb constituent from natural language text which relates two entities demands heavy linguistic processing. For the purpose of relation extraction by verb predicate, documents should be parsed into identified sentence structures.

For an example the sentence

"The greatest diversity of parrots is found in South America and Australasia ——— (1)

can be mapped to the above predicate format as follows after the sentence is tagged for syntactic constituents and concepts.

located_in(Parrot, South_America)
located_in(Parrot, Australasia)

In addition, some sentences give both positive and negative information with respect to a verb predicate.

For an example from the sentence

"Potoos are found in every Central and South American country except Chile." ——— (2)

We can extract the following relations.

located_in(Potoo, Central_American_country),
located_in(Potoo,South_American_country)
¬ located_in(Potoo, Chile)

But the negative relation cannot be identified from the syntactic structure and grammar rules only. Besides, negative information in a sentence may lead to extraction of false positive relations if the semantics of the negative language constructs are not dealt. Therefore we should pay attention to develop hypothesis to identify true relations and omit false relations. In our work we use a language parser to get the sentences syntactically analysed and further process the output produced by the parser in order to make them less complicated for accurate relation extraction.

IV. EXTRACTION OF RELATIONS FROM NATURAL LANGUAGE DEPENDENCIES

There are few natural language parsers available for the purpose of parsing texts. The Stanford parser[30] 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[31] 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. Stanford dependencies are binary relations held between a governor and a dependent. It represents all sentence relationships uniformly as typed dependency relations. The governors and dependents are the words in the sentence, represented in the relation with their position indicated by a number. These dependencies are quite effective in relation extraction. Another advantage of Stanford parser is that it gives basic and collapsed dependencies. In basic dependency representation each word

(except the head word of the sentence) in a sentence is the dependant of one other word. In the collapsed representation, dependencies involving prepositions, conjuncts, as well as information about the relative clauses are collapsed to get direct dependencies between the context words. This collapsing is often useful in simplifying patterns in relation extraction applications. First we use GATE to identify domain specific entities in a sentence and then use the parser on the annotated sentence.

For the sentence (1) the Stanford parser gives tagging and the collapsed dependencies given below.

The/DT, greatest/JJS, diversity/NN, of/IN, ***Parrots****/NNP, is/VBZ, found/VBN, in/IN,* ***South****/NNP,* ***America****/NNP, and/CC,* ***Australasia****/NNP*

det(diversity-3, The-1)
amod(diversity-3, greatest-2)
nsubjpass(found-7, diversity-3)
prep_of(diversity-3, ***Parrots****-5)*
auxpass(found-7, is-6)
root(ROOT-0, found-7)
nn(America-10, ***South****-9)*
prep_in(found-7, ***America****-10)*
prep_in(found-7, ***Australasia****-12)*
conj_and(America-10, ***Australasia****-12)*

Highlighted terms in dependencies indicate the already identified entities by the use of GATE and all the terms are syntactically tagged by the parser.

For the sentence (2) which gives both positive and negative relations parser outputs the following dependencies.

Potoos*/NNS, are/VBP, found/VBN, in/IN, every/DT, Central/NNP, and/CC, South/NNP, American/NNP, country/NN, except/IN,* ***Chile****/NNP*

nsubjpass(found-3, ***Potoos****-1)*
auxpass(found-3, are-2)
root(ROOT-0, found-3)
det(Central-6, every-5)
prep_in(found-3, Central-6)
nn(country-10, ***South****-8)*
nn(country-10, ***American****-9)*
prep_in(found-3, ***country****-10)*
conj_and(Central-6, country-10)
prep_except(found-3, ***Chile****-12)*

Both sentence 1 and sentence 2 give the same relation *located_in* and similar dependency clauses can be used to construct rules for relation extraction. The sentence (2) contains a negative dependency label *prep_except* which can be used to avoid extraction of negative relations.

Typed dependencies are also suitable for entity extraction though we have taken a different approach for entity extraction. In relation extraction, rules are generated to identify entity instances which are bound by a relation. If two entity instances in a sentence are not bound by an already identified relation, the relation verb is extracted to label the relation with new relation extraction rule. Similarly in entity extraction a noun or an adjective can be identified as an entity instance by using an appropriate extraction rule. Especially for the relation "*has_characteristic()*" which identifies the special characteristics of object or perhaps a living being, (e.g. hollow ball) the typed dependencies of the sentences annotated with object or person are searched for adjective clauses in order to extract the adjective that describes annotated entity. Similarly adverbs also play a significant role in identifying characteristics in the form of actions (e.g. run fast).

A. Reducing Typed Dependencies from Collapsed Dependencies

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. 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 contain at least one entity instance. We do not take into account the tense, the subject number or the voice of the sentence. All the nouns and verbs tagged originally by the parser are given category NN(stands for a noun) and VB (stands for verb in the base form) respectively. Therefore all the nouns and verbs annotated as NNS, NNP, VBS, VBP, VBG etc are simplified to either NN or VB. The syntactic categories and entity types are needed to be assigned to each term enclosed by dependency clauses in generating extraction rules. 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.

- label *"nsubj"* is the nominal subject which is the subject noun phrase and *"nsubjpass"* is the passive nominal subject in Stanford typed dependencies. Therefore the label *"nsubjpass"* is conveniently replaced by "*nsubj*". The main verb of the sentence is contained in "*nsubj*" or "*nsubjpass"* clauses.
- The label "*prep_including*" which enclose two instances of same entity is replaced by "*conj_and*" as both labels give similar dependencies. Two adjacent noun constituents or adjective and noun constituent in lables "*nn*" and "*prep_of*" are considered as one term on similar grounds.
- Adjective and noun contained in Adjectival modifier *"amod"* which is any adjectival phrase that serves to modify the meaning of the NP are considered as one term if both terms form a domain entity and the dependency *"amod"* is omitted from the background of the dependencies of the training sentences except for the relation "has_characteristic" . When "*amod*" encloses an adjective and a domain entity, it is the most significant clause which describes the characteristics of the entity. Therefore "*amod*" is not omitted from the dependencies of the test corpus. When joining the terms in the clauses *"nn"*, "*prep_of"* and "*amod*" , the clause "*nn*" is given the priority, then "*prep_of*" and last "*amod*"
- Since we consider only the relation verb which binds two entities, the clauses "*advcl*" and *"advmod"* which modifies a verb or the meaning of a verb can be ignored. But in the same time we can make use of these clauses as in the case with *amod* for the relations which

describe an action; for examples, *has_characteristic(Ostrich, run_fast)* or *run(Ostrich, fast)*.

- Auxiliary verbs *"aux"* and *"auxpass"* which are non main verbs of the clause such as "be", "have" etc. are ignored.
- 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 lableSS "*det*" and "*predet*" are ignored as it indicates the determinants.
- A dependency labeled as "*dep*" is ignored because dep is a very general dependency and used when the system is unable to determine a more precise dependency relation between two words.
- The clause "*cop*" which encloses a copular verb contributes a great deal for identification of taxonomic relations. But in non- taxonomic relations we can omit "*cop*" because a copular verb is not a significant part of the relation verb. But in non-taxonomies, when the clause "*nsubj*" does not enclose a verb constituent, copular verb is combined with the relevant noun in the clause in order to make the relation verb. Since the clause "*cop*" is needed in identifying taxonomic relations we do not remove "*cop*" from the dependencies of the sentences in extracting relations.

When a verb is joined with a noun the whole term is assigned the syntactic category *VB*.

Fig. 1 shows the process of generating reduced dependencies for a sentence from collapsed typed dependencies obtained from the Stanford parser.

The typed dependencies of the above mentioned sentences (1) and, (2) reduced according our criteria are shown below.

Sentence (1)

nsubj(found<VB>-7,
*greatest_diversity_of_**Parrots**<NN><**Bird**>-3)*
*prep_in(found<VB>-7,**South_America**<NN><**Location**>-10)*
*prep_in(found<VB>-7, **Australasia**<NN><**Location**>-12)*
*conj_and(**South_America**<NN><**Location**-10,*
*Australasia**<NN><**Location**>-12)*

Sentence (2)

*nsubj (found<VB>-3, **Potoos**<NN><**Bird**>-1)*
prep_in(found<VB>-3,
*South_American_country**<NN><**Location**>-10)*
conj_and(Central<NN>-6,
*South_American_country**<NN><**Location**>-10)*
*prep_except(found<VB>-3, **Chile**<NN><**Location**>-12)*

Sentences annotated with only one entity are treated for deriving taxonomical relations. Then the super-class of the annotated entity should be identified.

We consider *"Ostrich is a flightless bird."*
The Sentence leads to the extraction of relation *"is_a(Ostrich, flightless bird)"* from the following reduced typed dependencies.
*nsubj(bird<NN>-5, **Ostrich**<NN><Bird>-1)*
cop(flightless_bird<NN>-5, is<VB>-2)

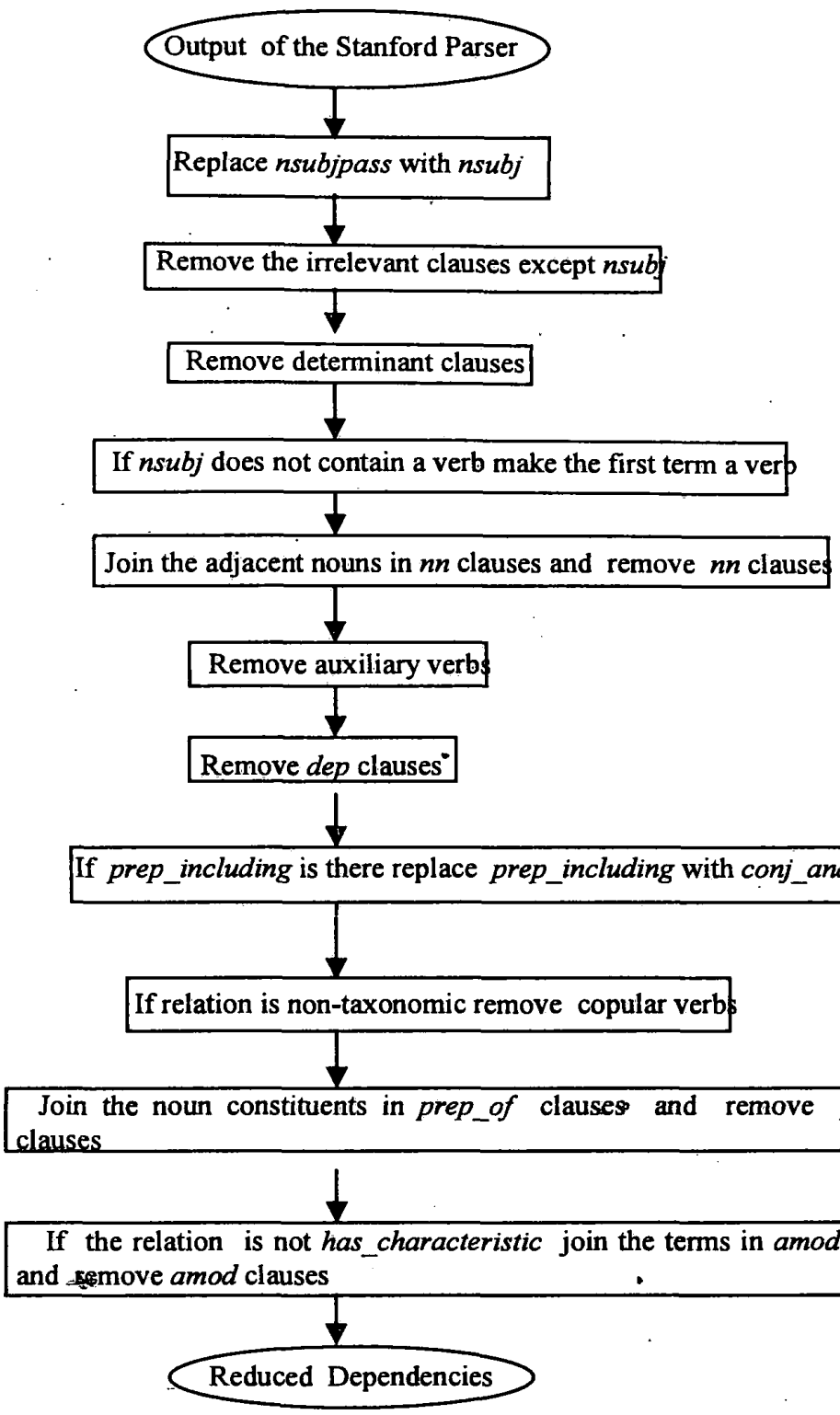

Fig. 1 The process of generating Reduced Typed Dependencies from Stanford Typed Dependencies of an English Sentence

Since we use supervised learning to extract relations from the reduced dependencies we use both positive and negative training examples. 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. USE OF AGENT TECHNOLOGY ON RELATION EXTRACTION FROM TYPED DEPENDENCIES

The Stanford typed dependencies and syntactic tags provide background knowledge to learn rules for relation extraction. We use an agent *OntoSupport*[32] to induce rules for relation extraction, searching through the typed dependencies of natural language sentences given in the training set[32]. Extraction rules for relations can be formulated as a combination of dependency labels and most common labels form a rule for a relation. The *nsubj* occurs in the typed dependencies of all the sentences. Therefore the simplest rule can be generated with *nsubj* and other most common label. We use a supervised learning method and hence provide a set of reduced typed dependencies of sentences annotated with domain specific entities as a training corpus for the rule learning process.

From the sentences that we have considered in our explanation for extraction of non taxonomical relations, we

can extract two positive and one negative relation instances for the relation *located_in()*. The label *prep_in* is also common for the reduced typed dependencies of both sentences in addition to *nsubj*. In formulating rules entity names are given priority over part of speech tags if a word is tagged as an entity.

Then *OntoSupport* can generate the unrefined preliminary rule

$$located\_in(Bird, Location) \leftarrow nsubj(VB, Bird) \wedge prep\_in(VB, Location)$$

All the entity instance pairs in reduced typed dependencies covered by a rule are taken as relation instances in order to make the extraction process more efficient and increase the coverage of the rule without missing any instances of an entity occurring in a sentence. Then we can represent above rule as predicate implication rule for the instances in the entity classes *Bird* and *Location* where $x \in \{Bird\}$ and $y \in \{Location\}$

$$\forall x \forall y \, ((nsubj(VB, x) \wedge prep\_in(VB, y)) \longrightarrow located\_in(x, y))$$

OntoSupport employs inductive logic programming technique (ILP)[33] [34] to derive the set of rules based on the text annotated with the entities.

In the rule learning process *OntoSupport* label a relation by the main verb constituent in the sentence. The main verb constituent of a sentence is normally wrapped in atomic formula *nsubj*. But when the sentence gets more complicated and contains more than one verb, identifying the relation verb of the annotated entities demand additional facts for correct identification of the relevant verb. In such situations the agent can make use of the position of the words in a sentence given by the parser. The minimum difference of the distance to both entities from each verb is considered as a measure in selecting the relation verb. When there is a situation of conflict where two verbs give the same difference value none of the verb is considered as a positive verb and the situation is left to be solved by human involvement.

When there are two "*nsubj*" atoms in the typed dependencies and two entities are divided in between two sections of *nsubj* the verb in the second section is taken as the positive verb.

The rules can be further enhanced with some resolutions to address the semantic ambiguity which is one of the difficulties that we come across in natural language processing. Two verbs can be considered as equivalent relation verbs (not synonyms) for a relation. For an example relation verbs "*live*" and "*found*" lead the way to the relation "*located_in*". Therefore "*live*" and "*found*" can be considered as equivalent terms for "*located_in*" under background information. *OntoSupport* keeps a set of equivalent verbs for a relation and updates the set whenever an equivalent term is found for the relation. Similarly from negative training data negative verbs for a relation can be identified and *OntoSupport* maintains set of negative verbs for relation in order to avoid the extraction of false positives. The verbs which give different relations between the same entities to the relation concerned, are also considered as negative verbs for the relation.

The agent *OntoSupport* learn rules to extract relation instances for a non-taxonomical relation such as *located_in, related, has_characteristics* etc as well as for taxonomical relations.

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 extract not only information for different users but can provide OntoSupport some information also in order to update its knowledge and rule set. We use JADE[35]; an agent framework to implement our agents.

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 (15 wikipedia web pages as training data and 27 wikipedia web pages as testing data). From the training data the *OntoSupport* learnt the rules shown in fig. 3 for the relation *located_in()* which exists between Bird and Country. While the agent is in action it is expected to learn more rules in the case of any deviation from the already created rules.

Based on the above mentioned training set *OntoSupport* first learns rules for pre defined non taxonomical relations *located_in(), related(), has_characteristics()* and the taxonomical relation *is_a()*. The relation *has_characteristics()* does not bind two entities by a verb. It binds the entities by an adjective and the adjective is joined with the corresponding entity to show the relationship as shown below by the set of rules generated for each relation where $x, t, s \in \{Bird\}$ and $y, z \in \{Location\}$, $p \in \{Part\}$

Relation: located_in(x, y)

$$\forall x \forall y \exists z \; ((nsubj(VB, x) \wedge conj\_and((VB \vee z), y) \wedge \neg prep\_from(VB, y) \wedge \neg prep\_for(VB, y) \wedge \neg prep\_except((NN, y) \wedge \neg negative(VB) \wedge \neg neg(VB, not)) \longrightarrow located\_in(x, y))$$

$$\forall x \forall y \exists z \; ((nsubj(VB, x) \wedge conj\_and(z, y) \wedge \neg prep\_from(VB, z) \wedge \neg prep\_for(VB, z) \wedge \neg prep\_except((NN, z) \wedge \neg negative(VB) \wedge \neg neg(VB, not)) \longrightarrow located\_in(x, z))$$

$$\forall x \forall y \; ((nsubj(VB, x) \wedge prep\_in(VB, y) \wedge \neg negative(VB) \wedge \neg neg(VB, not)) \longrightarrow located\_in(x, y))$$

$$\forall x \forall y \; ((nsubj(VB, x) \wedge prep\_on(VB, y) \wedge \neg negative(VB) \wedge \neg neg(VB, not)) \longrightarrow located\_in(x, y))$$

$$\forall x \forall y \; ((nsubj(VB, x) \wedge prep\_to(VB, y) \wedge \neg negative(VB) \wedge \neg neg(VB, not)) \longrightarrow located\_in(x, y))$$

Relation: related(x, t)

$$\forall x \forall t \exists s \; ((nsubj(VB, x) \wedge \wedge conj\_and((s, t) \wedge \neg conj\_only(NN, t) \wedge \neg negative(VB)) \longrightarrow related(x, t))$$

$\forall x \forall t$ ((nsubj(VB, x) $\wedge$ prep_to(VB, t) $\wedge$ $\neg$neg(VB, not) $\wedge$ $\neg$negative(VB)) $\longrightarrow$ related(x, t))

$\forall x \forall t$ ((nsubj(VB, x) $\wedge$ dobj(VB, t) $\wedge$ $\neg$cc(t, nor) $\wedge$ $\neg$negative(VB)) $\longrightarrow$ related(x,t)))

$\forall x \forall t$ ((nsubj(VB, x) $\wedge$ prep_of(VB, t $\wedge$ $\neg$negative(VB)) $\longrightarrow$ related(x,t))

Relation: has_characteristic(jj, p)

$\forall p$ ((nsubj(jj, p) $\wedge$ $\neg$prep_except_for(jj, p)) $\longrightarrow$ has_characteristic(jj, p))

$\forall p$ ((nsubj((VB $\vee$ p), (p $\vee$NN)) $\wedge$ amod(p, jj) $\wedge$ $\neg$neg(VB, VB)) $\longrightarrow$ has_characteristic(jj, p))

Taxonomic Relation
Relation is_a(x, jj_Bird)

$\forall x$ ((nsubj(jj_Bird, x) $\wedge$ cop(jj_Bird, VB) $\wedge$ $\neg$neg(x, not)) $\longrightarrow$ is_a(x, jj_Bird))

$\forall x$ ((nsubj(VB, jj_Bird) $\wedge$ prep_such_as(jj_Bird, x)) $\longrightarrow$ is_a(x, jj_Bird))

$\forall x \forall y$((nsubj(VB, jj_Bird) $\wedge$ prep_such_as(jj_Bird, x)) $\wedge$ appos(x,y)) $\longrightarrow$ is_a(y, jj_Bird))

$\forall x$ ((nsubj(include, jj_Bird) $\wedge$ prep_such_as(jj_Bird, x)) $\longrightarrow$ is_a(x, jj_Bird))

Where VB – Verb in Relation and negative(VB) – Verb is negative for the relation.

TABLE I: RELATIONS EXTRACTED BY ONTOEXTRACT

Pre-defined Relation	**Positive Equivalent Verbs**	**Negative Verbs**	**New Relations established by the Agent**
located_in(Bird, Location)	live, occur, are_native, found, colonise, establish, restricted	absent, extinct	farmed_in(Bird. Location) endangered_in (Bird, Location) is_national_bird (Bird, Location) worshipped_in (Bird, Location)
Related(Bird, Bird)	relate, share	unrelated, called, is_similar, associate, prey_for, is_called	associated_with(Bird, Bird) is_similar_to(Bird, Bird) prey_for(Bird, Bird) is_called(Bird, Bird)

OntoExtract applied the rules on 27 text documents and found relation instances for above mentioned relations.

Table I shows the positive and negative verbs with respect to two pre-defined non taxonomical relations and new relations established between the same entities in pre defined relations.

We have used the agent *OntoExtract* on the domains Bird to extract relation instances existing between annotated entities.

We have selected T. Wang and the team's approach [23] and R.C. Bunescu and R.J. Mooney's approach[27] for relation extraction to compare with the performance of *OntoExtract*. Since the availability of appropriate results for relation extraction is scarce in the literature, we could only choose the above mentioned approaches for our comparison despite the fact that it only categorizes relation instances to a number of pre defined relations. However both systems have considered a hierarchy of relations and used ACE training corpus. Bunescu and Mooney have used ACE 2002 whereas T.Wang and the team have used ACE 2004 with 7 main types and 22 sub types which is comparatively a higher number of pre-defined relations. Bunescu and Mooney present their evaluation based on two scenarios. We have picked the values relevant to the scenario which gives best overall performance. Comparison of OnoExtract's performance with the above mentioned systems is shown in Table II.

TABLE II: COMPARISON OF ONTOEXTRACT'S PERFORMANCE

System	Precision %	Recall %	F-Measure
OntoExtract	79.41	75.56	77.43
T.Wang's team Approach	73.87	69.5	71.59
Bunescu and Mooney's Approach with the CFG parser	65.5	43.8	52.5

Since our multi agent system does not leave any sentence annotated with entities and all the entity pairs in a covered sentence are taken as relation instances without distinguishing between entities, we can achieve a higher recall. Consideration of both positive and negative data in generating rules contributes for a higher precision.

Although we have used a smaller number of training examples to initiate the system, it will not affect the performance of the system because any situation that cannot be covered by the extraction rules is considered as an instance for a new relation and a new extraction rule is generated for the relation accordingly. Therefore use of a smaller training data set becomes an advantage here and has no adverse effect on the performance of the entire system. Since the agent's knowledge is updated continuously with additions to the set of rules and set of equivalent verbs while the agents are in action we can achieve a higher performance measures.

VII CONCLUSIONS

In this paper we have discussed how typed dependencies of a sentence make a suitable candidate for relation extraction. We have shown the way that typed dependencies are pruned to reduce the number of dependency clauses of a sentence,

eliminating unnecessary dependencies. A set of rules for relation extraction is learnt from the reduced typed dependencies of training data(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. The rules generated by the agent *OntoSupport* are represented in the predicate implication form in order to find the relations between all the entity instances in a sentence. 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. While the agents are progressing in the system, the existing rules will also be subjected to further refinement. The same set of rules can be tried in 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 sentences.

ACKNOWLEDGMENT

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

REFERENCES

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

[2] 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.

[3] 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," in *Proc. Fifteenth National Conference on Artificial Intelligence*, 1998.

[4] 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.

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

[6] 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.

[7] M. Dzbor, J. Domingue, E. Motta, "Magpie-toward a semantic web browser," in *Proc. of the International Semantic Web Conference*, 2003.

[8] B. Popov, A. Kiryakov, D. Ognyanoff, D. Mahov, A. Kirilov and M. Goranov ."Towards semantic web information extraction". *Human Language Technologies Workshop at the 2nd International Semantic Web Conference.*, October 2003. [Online]. Available: http://gate.ac.uk/conferences/iswc2003/proceedings/popov.pdf

[9] A. Wasilevska "Apriori Algotithm" Available: http://www.icaen.uiowa.edu/~comp/Public/Apriori.pdf

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

[11] D. Celjuska., M. Vargas-Vera., "Ontosophie A Semi-automatic system for ontology population from text", *International Conference on Natural Language Processing,* 2004.

[12] 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.

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

[14] 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.

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

[16] H. Cunningham., D. Maynard., K..Bontcheva,. and V.Tablan , "GATE: A framework and graphical development environment for robust NLP tools and applications.," in *Proc. of the 40th Anniversary Meeting of the Association for Computational Linguistics,* pp 168-175, 2002

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

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

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

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

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

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

[23] T. Wang, K. Bontcheva, Y. Li, H. Cunningham and J. Wang "Automatic extraction of hierarchical relations from text," in the *The Semantic web; Research and Applications, 3rd European Semantic Web Conference, ESWC*, 4011 in lecture Notes in Computer science pp. 215-229, Springer, 2006.

[24] Surpport Vector Machines. Available: http://en.wikipedia.org/wiki/Support_vector_machine.

[25] http://www.itl.nist.gov/iad/mig/tests/ace/2004/

[26] A. Culotta and J. Sorensen, "Dependency tree kernels for relation extraction," in *Proc. of the 42nd Annual Meeting of the Association for Computational Linguistics(ACL-54),* Barcelona, Spain, July 2004.

[27] R. C. Bunescu and R. J. Mooney, "A shortest path dependency kernel for relation extraction, in *Proc. of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, pp 724-731, Vancouver, October, 2005.

[28] K. Fundel, R. Kuftner and R. Zimmer, "Relation extraction using dependency parse trees," in *Bioinformatics Advanced Access*, December 2006.

[29] A. Akbik and J. BroB, "Wanderlust: Extracting semantic relations from natural language text using dependency grammar patterns," in Proc. of the *The WWW 2009 Workshop on Semantic Search*, pp 6-15, Madrid, Spain, 2009

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

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

[32] 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*,, pp 70-76, Chengdu, China, June 2011

[33] M. D. S. Seneviratne and D. N. Ranasinghe, "Inductive Logic Programming in an Agent System for Ontological Relation Extraction," *International Journel of Machine Learning and Computing, Vol. 1 No. 4*, pp 344-352, October 2011

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

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