Discovering Relations between Noun Categories

Thahir P Mohamed [*]
University Of Pittsburgh
pmthahir@gmail.com

Estevam R Hruschka Jr.
Federal University of Sao Carlos
estevam@cs.cmu.edu

Tom M Mitchell
Carnegie Mellon University
tom.mitchell@cs.cmu.edu

Abstract

Traditional approaches to Relation Extraction from text require manually defining the relations to be extracted. We propose here an approach to automatically discovering relevant relations, given a large text corpus plus an initial ontology defining hundreds of noun categories (e.g., Athlete, Musician, Instrument). Our approach discovers frequently stated relations between pairs of these categories, using a two step process. For each pair of categories (e.g., Musician and Instrument) it first co-clusters the text contexts that connect known instances of the two categories, generating a candidate relation for each resulting cluster. It then applies a trained classifier to determine which of these candidate relations is semantically valid. Our experiments apply this to a text corpus containing approximately 200 million web pages and an ontology containing 122 categories from the NELL system [Carlson et al., 2010b], producing a set of 781 proposed candidate relations, approximately half of which are semantically valid. We conclude this is a useful approach to semi-automatic extension of the ontology for large-scale information extraction systems such as NELL.

1 Introduction

The Never-Ending Language Learner (NELL) (Carlson et al., 2010b)) is a computer system that learns continuously to extract facts from the web. NELL is given as input an initial ontology that specifies the semantic categories (e.g. city, company, sportsTeam) and semantic relations (e.g. hasOfficesIn(company,city), teamPlaysInCity(sportsTeam,city)) it must extract from the web. In addition, it is provided 10-20 seed positive training examples for each of these categories and relations, along with hundreds of millions of unlabeled web page. Given this input, NELL applies a large-scale multitask, semisupervised learning method to learn to extract new instances of these categories (e.g., city("London")) and relations (e.g., teamPlaysInCity("Steelers","Pittsburgh")) from the web. During the past 17 months NELL has been running nearly continuously, learning to extract over 600 categories and relations, and populating a knowledge base containing over 700,000 instances of these categories and relations with a precision of approximately 0.85[1].

This paper considers the problem of automatically discovering new relations to extend the ontology of systems such as NELL, enabling them to increase over time their learning and extraction capabilities. More precisely, we consider the following problem:

Input:
· An ontology specifying a set of categories
· A knowledge base containing instances of these categories (perhaps including errors)
· A large text corpus

Output:
· A set of two-argument relations that are frequently mentioned in the text corpus, and whose argument types correspond to categories in the input ontology (e.g., RiverFlowsThroughCity(<River>,<City>).
· For each proposed relation, a set of instances (i.e. RiverFlowsThroughCity("Nile","Cairo")).
· For each proposed relation, a set of text extraction patterns that can be used to extract additional instances of the relation (e.g., the text "X in the heart of Y", where X is a known river, and Y a known City, suggests extracting RiverFlowsThroughCity(X,Y)).

Note the above inputs are easily available from NELL in the form of its existing ontology and extracted knowledge base. Note also that the outputs

* Thahir P. Mohamed is currently at Amazon Inc.
1 NELL's extracted knowledge can be viewed and downloaded at http://rtw.ml.cmu.edu.

of our system are sufficient to initiate NELL's learning of additional extraction methods to further populate each proposed relation. One goal of this research is to create a system that can provide NELL with an ongoing set of new learning and extraction tasks. The system is called OntExt (Ontology Extension System)

Table 1 shows a sample of successful relations and corresponding relation contexts and sample seed instances generated by OntExt.

Table 1. Examples of valid relations (generated by OntExt), their text extraction patterns and extracted instances.

name(category1-main context-category2)	Extraction patterns	Seed Instances
River -in heart of- City	'in heart of' 'in the center of' 'which flows through'	"Seine, Paris" "Nile, Cairo" "Tiber river, Rome" "River arno, Florence"
Food -to produce- Chemical	'to produce' 'to make' 'to form'	"Salt, Chlorine" "Sugar, Carbon dioxide" "Protein, Serotonin"
StadiumOrVenue -in downtown- City	'in downtown'	"Ford field, Detroit" "Superdome, New Orleans" "Turner field, Atlanta"
Disease -caused by- Bacteria	'caused by' 'is the causative agent of' 'is the cause of'	"pneumonia, legionella" "mastitis, staphylococcus aureus" "gonorrhea, neisseria gonorrhoeae"
Disease -destroys- CellType	'destroys' 'attacks'	"alzheimer, brain cells" "vitiligo", melanocytes" "aids, lymphocytes"
County -county- StateOrProvince	'county' 'county of' 'county in'	"sufolk, massachusetts" "marin, california" "sussex, delaware" "osceola, michigan"

2 Background

Traditional Relation Extraction

We define Traditional RE systems as those that require the user to specify information about the relations to be learned. For instance, SnowBall (Agichtein and Gravano 2000) & CPL (Carlson et al. 2009) are bootstrapped learning systems that require manual input of relation predicates. In these systems, for each relation predicate, the relation name (e.g. City 'Capital of' Country), the seed instances and the category type (e.g. City, Country, Celebrity etc) are provided (for domain and range). In CPL (Carlson et al. 2009), learning of relation/category instances is coupled by using constraints such as mutual exclusion relationships among the predicates. The authors show that this coupling reduces semantic drift, which commonly occurs with bootstrapping systems, thus leading to improved precision. CPL achieved 89% precision for the relation instances extracted (Carlson, Betteridge et al. 2009). KNOWITALL (Etzioni, Cafarella et al. 2005) is a web-scale relation extraction system, which requires as input the relation names. Hence, in these "traditional relation extraction" methods, the need to manually define the relations to be extracted makes it difficult to work in applications having thousands of possible relation predicates.

2.1 Open Relation Extraction

Open RE methods do not require a user to manually specify the information about the relations to be learned, such as their names, seed examples, etc. TextRunner (Banko, Cararella et al. 2007) is such an Open Information Extraction system that retrieves from the web millions of relational tuples between noun phrase entities. TextRunner uses a deep linguistic parser to perform self-supervised learning and extracts a positive set (i.e. valid relation between entities) and a negative set (i.e. invalid relationships) of relational tuples based on certain heuristics. Then, a Naive Bayes classifier is built having features such as part-of-speech tags of the words in the relation tuples, number of tokens, stopwords etc., and uses the labeled instances as the training set. This classifier runs on sentences from a web corpus to extract millions of relational tuples. However, of the 11 million high confident relational tuples extracted by this system only 1 million were concrete facts (Banko, Cararella et al. 2007). Of these concrete facts 88% were estimated to be correct. For instance, (Mountain View, headquarters of, Google) is a tuple representing a valid concrete fact. The remaining 90% of the tuples are abstract or do not have well-formed arguments or well-formed relations. For instance, (Einstein, derived, theory) is an abstract tuple as it does not have enough information to indicate a concrete fact (Banko, Cararella et al. 2007) because the specific theory which Einstein derived is missing in that tuple. In the tuple (45, 'went to', 'Boston'), one of the arguments (i.e. 45) is not well formed.

In (Banko and Etzioni, 2008) a Conditional Random Field (CRF) classifier is used to perform Open Relation Extraction which improves by more than 60% the F-score achieved by the Naive Bayes model in the TextRunner system. However the CRF approach does not solve the problem associated with extraction of abstract/non-well formed tuples. Further, in the same work, it is shown that Open RE has a much lower recall in comparison to Traditional RE systems. On four common relations (Acquisition, Birthplace, InvetorOf, WonAward), Open RE attained a recall of 18.4% in comparison to 58.4% achieved by Traditional RE (Banko and Etzioni 2008). Both Open RE systems discussed (Banko, Cararella et al. 2007; Banko and Etzioni 2008) do not perform learning of the category type of the entities involved in the relations. They are single-pass and do not perform continuous learning to improve/extend on what has been learnt.

2.2 Unsupervised Methods to Extract Relations between Named Entities

In general, traditional RE methods extract concrete facts and have much higher recall for a given relation, than Open RE methods. This is due to the knowledge fed into Traditional RE methods such as the category type of the entities in the relation and seed instances for the relation. Traditional RE methods require the relations to be manually defined and extract instances only for them. Open RE methods, on the other hand, do not require any such domain specific knowledge to be manually input. They extract instances for a wide spectrum of relations that are not manually pre-defined.

To overcome the drawbacks of using Traditional and Open RE methods, some researchers have used unsupervised learning methods to automatically generate new relations (with seeds and contexts) between specific categories. These automatically generated relations can then be used as input to Traditional RE systems.

Hasegawa et.al (Hasegawa, Sekine et al. 2004), propose an unsupervised clustering based approach. One feature vector for each co-occurring NE pair is formed based on the context words in which the NE pair co-occurs. Then, a cosine-similarity metric is applied to each pair of feature vectors to generate a "NE-pair x NE-pair" matrix. Clustering is done on this matrix and each cluster of NE-pairs corresponds to a relation predicate.

The work by Zhang et.al (Zhang, Su et al. 2005) generates a shallow parse tree for each sentence containing a NE pair to generate relation instances. A tree similarity metric is used to cluster the relation instances. This method gives improved F-score over Hasegawa et.al (Hasegawa, Sekine et al. 2004). Further they use a specialized NE tagger built to recognize entities that belong to specific predefined categories. The aforementioned methods (Hasegawa, Sekine et al. 2004) (Zhang, Su et al. 2005) were tested on a news corpus to identify relations between only a couple of pairs of entity types (Person-GeoPoliticalEntity and Company-Company).

Both of these methods cluster NE-pairs primarily based on lexical similarity of the context words connecting the entities. Hence NE-pairs connected by lexically different but semantically similar context patterns (e.g. river 'in heart of' city and river 'flows through' city) would probably not get clustered together. The web data is, however, much noisier and has a larger number of entity types (i.e. category predicates), thus, another issue is that for web scale data *NE pairs X NE pairs* similarity matrix would not be scalable for many thousands of NE-pairs.

3 Ontology Extension System - OntExt

The OntExt system for ontology extension, proposed in this paper, combines characteristics from both "Traditional RE" and "Open RE," to discover new relations among categories that are already present in the ontology, and for which many instances have already been extracted.

Our proposed method for automatic relation extraction offers the following advantages over the methods discussed above.

- The key idea in our approach is to make use of redundancy of information in web data - the same relational fact is often stated multiple times in large text corpora, using different context patterns. We use this redundancy to cluster together context patterns which are semantically similar although they may be lexically dissimilar.

- Instead of clustering on the *'NE-pairs X NE-pairs'* matrix, clustering is done on a *'Context-pattern X Context-pattern'* matrix. This is much more scalable as the context patterns are fewer in number and since our method applies several criteria to prune out irrelevant patterns.
- To accommodate errors in the input category instances and ambiguity in web data, we build a classifier which learns to distinguish valid relations from semantically invalid relations.

OntExt has 3 components. 1) It starts exploring a large web corpus and 2) category instances extracted by CPL to generate new relations. After the relations are generated, 3) a classifier is developed to classify semantically valid relations.

3.1 Pre-processing

Following along the same strategy used in [Carlson et al., 2010], OntExt uses as input a corpus of 2 billion sentences, which was generated by using the OpenNLP[2] package to extract, tokenize, and POS-tag sentences from the 500 million web page English portion of the ClueWeb09 data [Callan and Hoy, 2009]. Before performing relation extraction, this corpus is preprocessed. First, sentences which contain a pair of known category instances are retrieved (e.g. the sentence "Ottawa is the capital of Canada.", where 'Ottawa' is a known instance of the 'City' category and 'Canada' is a known instance of 'Country'). For every category pair (e.g. <City, Country>) the sentences containing known instances of both categories are grouped into a set S. The text between the two instances is called the 'context pattern' (e.g. 'is the capital of' is a context pattern). Three types of pruning are done on this set S.

1. If the context pattern is a rare one (i.e. if the context pattern occurs in less than a threshold number of sentences), all sentences with that context pattern are removed. Thus we retain only frequently occurring contexts. We use a threshold requiring at least 5 sentences in the experiments presented in Section 4.
2. Context patterns which co-occur with very few instances of either category type are removed. For example, the category pair <Vehicle,SportsTeam> has several sentences such as 'Car was engulfed in flames', 'Truck was engulfed in flames' etc. Note that Flames (Calgary Flames) is a SportsTeam. But here flames clearly does not refer to a Sportsteam. This context 'was engulfed in' connects several instance of a 'Vehicle' category to a single instance of SportsTeam instance. Hence all sentences with this context are removed. Note this context would not have been removed in step 1 as that is just a threshold on the number of sentences in which any pair occurs. We use a threshold requiring at least 3 distinct instances of both the domain and the range, for each context.
3. Banko et.al, 2008 show that most binary relational contexts fall under certain types of lexico-synctatic patterns. They include context patterns like 'C1 Verb C2', 'C1 NP Prep C2', 'C1 Verb Prep C2' and 'C1 to Verb C2' (C1 and C2 are category instances). Hence context patterns which do not fall under the above types are removed from the set S as they are not likely to produce relation instances.

3.2 Relation Generation

From the previous pre-processing step OntExt retrieves for each category pair a pruned set S' of sentences. Each sentence has a pair of category instances and the context connecting them.

Algorithm 1: Relation Generator

Input: One pair of Categories (C1, C2) and set of sentences, each containing a pair of instances known to belong to C1 and C2. The phrase connecting the instances in the sentence is the context.
Output: Relations and their seed instances

Steps:
1. From the input sentences, build a Context by Context co-occurrence matrix (Shown in figure 1). The matrix is then normalized.
2. Apply K-means clustering on the matrix to cluster the related contexts together. Each cluster corresponds to a possible new relation between the two input categories. (Weka Machine Learning package [Hall et al., 2009] was used to perform K-means clustering. The value of K was set to 5 based on trial and error experiments.)

2 http://opennlp.sourceforge.net.

3. Rank the known instance pairs (belonging to C1,C2) for each cluster and take the top 50 as seed instances for the relation

The key data structure used by OntExt is a co-occurrence matrix of the contexts for each category pair, as shown in Figure 1. In this matrix, each cell corresponds to the number of pairs of category instances that both contexts co-occur with (e.g. the sentences "Vioxx can cure Arthritis" and "Vioxx is a treatment for Arthritis" provide a case where the 2 contexts 'can cure' and 'is a treatment for' co-occur with an instance pair [Vioxx, Arthritis]). Initially, the value of Matrix(I,j) is the number of category instance pairs that occur with both context i and context j. We then normalize each cell in the matrix, dividing it by by the total count for its row.

$$Matrix(i,j) = \frac{Matrix(i,j)}{\sum_{j=0}^{N} Matrix(i,j)}$$

We also give higher weight to contexts which co-occur with only a few contexts over ones which are generic and co-occur with most contexts.

$$Matrix(i,j) = Matrix(i,j) * \frac{N}{|\{Context(j) : Matrix(i,j) > 0\}|}$$

Where N is the total number of contexts, and |{Context(j) : Matrix(i,j) > 0}| refers to the number of cells in the row Matrix(i) which are greater than zero.

For example, for the <drug, disease> category pair after 122 contexts were obtained after preprocessing. Contexts such as 'to treat', 'for treatment of', 'medication' which all indicate the same relation (drug-to treat-disease) have high co-occurrence values (see Figure 1). Similarly contexts such as 'can cause', 'may cause', 'can lead to' (indicating the relation drug-can cause-disease) have high co-occurrence values (see Figure 1). When OntExt performs clustering on this co-occurrence matrix the contexts with large co-occurrences get clustered together. Each cluster is then used to propose a possible new relation. The centroid of each cluster is used to build the relation name. If the centroid of a cluster is the context 'for treatment of', then the relation name is 'drug-for-treatment-of-disease'.

OntExt next generates seed instances for the proposed relation. The seed instances which co-occur with contexts corresponding to the cluster centroid or close to centroid will be best representative of the relation. So the strength of the seed instance is inversely proportional to the standard deviation of the context from the centroid of the relation contexts cluster. Also the strength of the seed instance is directly proportional to the number of times it co-occurs with the context.

Contexts/ Contexts	may cause	can cause	can lead to	to treat	for treatment of	medication
may cause	0.176	0.074	0.030	0.015	0.011	0.000
can cause	0.051	0.150	0.039	0.018	0.013	0.010
can lead to	0.034	0.064	0.189	0.019	0.021	0.018
to treat	0.006	0.011	0.007	0.109	0.043	0.015
for treatment of	0.005	0.008	0.008	0.045	0.086	0.023
medication	0.000	0.011	0.009	0.030	0.036	0.111

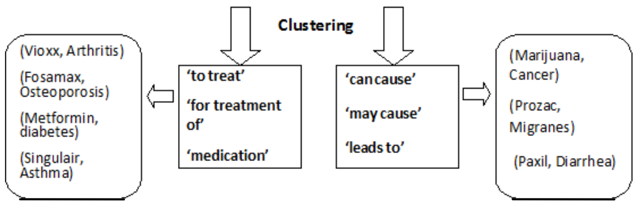

Figure 1: This figure shows the Context by Context sub-matrix (with 6 contexts) for the category pair (Drug, Disease) and the seed instances for each relation. As described in the text, each entry gives the normalized count of the number of known <drug, disease> pairs that occur with both the row context and the column context.

To summarize, each seed instance *s* (pair of category instances) is weighted as follows

$$\sum_{c \in Pattern\ cluster} Occ(c,s)/(1+sd(c))$$

Where,
Pattern_cluster is the cluster of pattern contexts for this given relation
Occ(c,s) is the number of times instance 's' co-occurs with the pattern context 'c'
sd(c) is the standard deviation of the context from the centroid of the pattern cluster.
Using this metric the instances are ranked and the top 50 are output as initial seed instances for the proposed relation.

3.3 Classifying semantically valid relations

More than half of the relations generated in the previous step are invalid due to the following reasons

1. Error in category instances: The category instances input to OntExt come from NELL. In the version of the knowledge base used in these experiments, the accuracy of these instances was 78%. Due to the erroneous category instances some invalid relations are generated by OntExt. For instance the generated relation, 'condiment-wearing-clothing' with seeds (pig,dress), (rabbit,pants) etc. Here 'pig' and 'rabbit' were incorrectly identified by NELL as instances of 'condiment'.
2. Semantic Ambiguity: Consider the generated relation 'bakedgood-baking-magazine' with instances (cookies,time), (cupcakes, people), etc. Here the instances 'time' and 'people' do not refer to magazines, although they can in general. Due to the semantic ambiguity of these instances this invalid relation got generated
3. Semantically Incomplete relations: Some of the generated relations require a third entity or some more contextual information, in order to be considered semantically valid. For instance, 'personUs-said-company' or 'newspaper-is-reporting-that-company'. These don't stand by themselves as two-argument relational facts and need more information to be complete
4. Illogical relations: Some generated relations simply have no real semantic meaning. These relations are generated due to the category instances appearing together in some unrelated contexts. E.g. the generated relation 'date-starting-date' with seeds such as (Wednesday, June), (friday, July) and the relation 'country-minister-of- economicsector' with seeds (japan,agriculture), (india, industry).

The introduction of these invalid relations can adversely affect the performance of NELL. However, it is a challenging problem to develop automated ways to distinguish between valid and invalid relations without any domain specific knowledge. To approach this problem, we identified a set of features which can help characterizing valid and invalid relations, and which can be generated automatically. Below is a description of the features and the intuition behind their use for this classification task.

Each generated relation has a pair of category types (C1, C2), a corresponding set of seed instances (which are pairs of instances belonging to C1 and C2) and pattern contexts connecting C1 and C2. Let N be the number of seed instance pairs and N1 and N2 be number of unique instances (out of these N instance pairs) belonging to categories C1 and C2 respectively.

1. Normalized frequency count: The frequency count of each category instance is obtained from the corpus and normalized by the category instance with maximum count. For a given relation, a feature is generated by averaging the normalized frequency counts of the instances belonging to C1. Another similar feature is generated for C2 following the same strategy. For example the relation <Profession 'believe that' Movie> was generated due to common words like 'predator', 'earthquake' being identified as movie names out of context. These features can help identify such invalid relations.
2. Distribution of extraction patterns: NELL learns instances as well as extraction patterns for each category (e.g. the category Actor has extraction patterns such as '_ got an Oscar award', '_ is the movie's lead actor'). If a category instance co-occurs in the web corpus with several extraction patterns belonging to other categories, then that instance has large ambiguity. We measure ambiguity of an instance (i) belonging to category 'C' with respect to another category 'M' (where M is not a sub type or super type of 'C') as
   Ambiguity(i,M) =
   $$\frac{\#\text{of extraction patterns in 'M' that 'i' co-occurs with}}{\#\text{of extraction patterns in 'C' that 'i' co-occurs with}}$$
   We measure the average ambiguity for the set of instances (of size N) belonging to category C in the generated seeds as follows,
   $$Max_M\left(\sum_{i \in C} Ambiguity(i,M)/N\right)$$
   Two features are generated for categories C1 and C2 in the relation.
3. Relationship characteristics: We identified a few characteristics of the relation which help in identifying valid relations. If in the generated relation, most instances of C1 co-occur only

with very few instances of C2 (or vice versa) then the relation could be weak. For example, <Organization 'Provides' EconomicSector> - the instance 'Information' (of category EconomicSector) connects to a large percentage of items in the category 'Organization' but does not express a meaningful relation. So we consider the instance (in this example 'Information', let us call it 'maxconnect_instance') co-occurring with maximum number of instances of the other category. The percentage of instances it co-occurs with from among the total number of instances of the other category which are part of the seed instances is taken as a feature. Also if that instance is a very common word (like 'information' which in several contexts does not refer to 'EconomicSector') then this could indicate the presence of an invalid relation. So the normalized frequency count of this instance (maxconnect_instance) is taken as another feature.

4. Pattern Contexts: The number of pattern contexts attained through pattern clustering for the relation is taken as another feature. The presence of several pattern contexts connecting the instances between the two categories could indicate that the relation is a valid one. The presence of Hearst patterns (Hearst M, 1992) referring to a hyponym ("is-a") relation in pattern contexts indicates the possibility of a valid relation, and is taken as another feature

   Another feature is regarding how specific is the context pattern to this relation. If the same context connects say C1 instances to instances of several other categories apart from C2, then this context is not unique to this relation and might not indicate a meaningful valid relationship. So the ratio of the number of instances in C2 connected to C1 versus the number of instances from all categories connected to C1 by the most significant pattern context (i.e. centroid in pattern cluster) is taken as a feature. A similar feature is generated for C2 as well.

4 Experimental Setup and Results

4.1 CPL System

CPL (Carlson et al., 2010) is a semi-supervised learning system which takes in an input ontology (containing category and relation predicates and corresponding seed instances) and constraints (such as Mutual exclusion rules between predicates). The system iteratively extracts patterns and instances for the category/relation predicates from a web corpus of around 500 million web pages. CPL is one learning component in NELL (the Never Ending Language Learner) (Carlson et al., 2010b).

4.2 Relation Generation:

We use approximately 22,000 category instances belonging to 122 categories extracted by CPL at the end of its 20th iteration and the web corpus as input to perform the co-clustering described in Section 3.2 and generate the new relations. The process generated 781 relations. For each relation, the relation name, types of the categories involved in the relation and the seed instances and patterns for each relation were generated. Table 1 in section 1 shows a sample of valid relations generated by this method.

Tables 2, 3, 4 and 5 show invalid relations for each type of invalidity, "Error in the Category Instances", "Semantic Ambiguity", "Semantically Incomplete Relations" and "Illogical Relations" respectively. More specifically, Table 2 shows a sample of relations generated due to an entity being labeled incorrectly as to belong to a category. The incorrect category instances are in italics.

Table 3 presents a sample of relations which were generated because of semantic ambiguity. Instances with ambiguity are in italics.

Table 4 shows some of the generated relations which are semantically incomplete.

Table 5 presents samples of illogical relations which do not establish any concrete fact.

Table 2. Examples of Incorrect category instances.

name(category1 -main context-category2)	Relation Contexts	Seed Instances
SportsGame -Beating-Country	'beating'	"tournament,Sri Lanka" "champions, France" "match, canada"
Animal -will eat-Condiment	'will eat' 'eating'	"wolf, sheep" "fox, rabbit" "lion, lamb"

Table 3. Examples of Semantically Ambiguous relations.

Name	Relation Contexts	Seed Instances
Bird -play- City	'play'	"Cardinals, Atlanta" "Ravens, Miami" "Eagles, Chicago"
BakedGood -baking- Magazine	'baking'	"time, cakes" "people, cookies"

Table 4. Examples of semantically incomplete relations.

Name	Relation Contexts	Seed Instances
Personus acknowledged Date	'acknowledged' 'warned' 'met'	"mr obama, tuesday" "george w . bush, tuesday" "al gore, thursday"
NewsPaper -is reporting that- Company	'is reporting that' 'writes that' 'reported that'	"financial times, apple" "wall street journal, gm" "wall street journal, yahoo"

Table 5. Examples of relations representing facts that are not concrete.

Name	Relation Contexts	Seed Instances
Emotion -of living in- StateOrProvince	'of living in'	"joy, california" "excitement, colorado" "fear, iowa"
BodyPart -to keep- BodyPart	'to keep' 'guard'	"hand, eye" "nose, throat" "eye, brain" "elbow, hand"

4.3 Relation Classification:

To determine the feasibility of automatically classifying OntExt's proposed relations as valid or invalid, we trained and tested a classifier using the features described above, using manually assigned class label for some of the generated relations (252 relations) as valid or invalid (the criteria for which was explained before). 115 of these 252 relations were found to be valid by manual evaluation. This shows the need for a machine learning classifier to identify valid/invalid relations. The various features described earlier (such as normalized frequency count, relationship characteristics, pattern context features, distribution of extraction patterns) were generated for each relation. Ten-fold cross validation experiments were carried out with various classifiers. A Random Forest classifier performed the best. Precision, recall and ROC-area is shown in the table below (ROC area is the area under the ROC curve which plots the classifier performance by having the True Positive Rate on the Y-axis and False Positive Rate on the X-axis).

Table 6. Classifier performance.

RelationType	Precision	Recall	ROC Area
Valid	71.6	72.2	0.804
Invalid	76.5	75.9	0.804
Weighted Avg.	74.2	74.2	0.804

These results indicate that the system is able to learn to identify semantically valid relations without using any manually input information. The valid relations generated can be input to NELL, allowing it to iteratively learn additional instances for each proposed relation.

5 Conclusion and Future work:

Open Relation Extraction and Traditional Relation Extraction have their respective strengths and weaknesses. The OntExt system proposed in this work combines the strengths of both of those methods. The relation predicates automatically generated by our approach are typed, have a meaningful name identifying the relation, and are accompanied by suggested context patterns and seed instances. These relations can be input to NELL to learn more instances for the relation. We propose in the future to integrate this relation generation system into NELL, to iteratively extend NELL's initial ontology, providing an ongoing stream of new learning tasks. After every fixed set of NELL's iterations, its growing knowledge base would be input to the relation generation system which will in turn feed NELL with new relation predicates. One additional area for future research is to extend OntExt to discover new categories in addition to new relations.

Acknowledgements

We gratefully acknowledge support for this research from Darpa, Google, Yahoo! and the Brazilian research agency CNPq. We also gratefully acknowledge Dr. Madhavi Ganapathiraju (at University of Pittsburgh) for her support and encouragement.

References

Agichtein, E. and L. Gravano (2000). "Snowball: Extracting relations from large plain-text collections." Procs. of the Fifth ACM International Conference on Digital Libraries.

Banko, M., M. Cararella, et al. (2007). "Open information extraction from the web." In Procs. of IJCAI.

Banko, M. and O. Etzioni (2008). "The Tradeoffs Between Open and Traditional Relation Extraction." In Proceedings of ACL-08.

Callan, J., and Hoy, M. (2009). Clueweb09 data set. http://boston.lti.cs.cmu.edu/Data/clueweb09/.

Carlson, A., J. Betteridge, et al. (2009). "Coupling Semi-Supervised Learning of Categories and Relations." Proceedings of the NAACL HLT 2009 Workshop on Semi-supervised Learning for Natural Language Processing.

A. Carlson, J. Betteridge, et al. (2010). "Coupled Semi-Supervised Learning for Information Extraction," Proceedings of the ACM International Conference on Web Search and Data Mining (WSDM), 2010.

A. Carlson, J. Betteridge, et al., (2010b). "Toward an Architecture for Never-Ending Language Learning," Proceedings of the Conference on Artificial Intelligence (AAAI), 2010.

Etzioni, O., M. Cafarella, et al. (2005). "Unsupervised named-entity extraction from the web: An experimental study." Artificial Intelligence.

Hasegawa, T., S. Sekine, et al. (2004). "Discovering relations among named entities from large corpora." Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics

Zhang, M., J. Su, et al. (2005). "Discovering Relations between Named Entities from a Large Raw Corpus Using Tree Similarity-based Clustering." IJCNLP 05.

Hasegawa, T., S. Sekine, et al. (2004). "Discovering relations among named entities from large corpora." Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics

Zhang, M., J. Su, et al. (2005). "Discovering Relations between Named Entities from a Large Raw Corpus Using Tree Similarity-based Clustering." IJCNL

Hearst, M. (1992) Automatic Acquisition of Hyponyms from Large Text Corpora. Proc. of the Fourteenth International Conference on Computational Linguistics, Nantes, F

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten (2009); The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1.