User-driven Relational Models for Entity-Relation Search and Extraction

Jay Urbain
Milwaukee School of Engineering
1025 N. Broadway Ave.
Milwaukee, WI 53202
1-414-745-5102
urbain@msoe.edu

ABSTRACT

The ability to extract new knowledge from large datasets is one of the most significant challenges facing society. The problem spans across domains from intelligence analysis and scientific research to basic web search. Current information extraction and retrieval tools either lack the flexibility to adapt to evolving information needs or require users to sift through search results and piece together relevant information. With so much data compounded by the criticality of finding relevant information, new tools and methods are needed to discover and relate relevant pieces of information in ever expanding repositories of data.

We posit that user-driven relational models are needed to collectively learn and discover fine-grained entities and relations that are relevant to a user's information need. To meet this need, we present a ranked retrieval and extraction framework for collectively learning and integrating evidence of entities and relational dependencies to predict at query time, a ranking of sentences containing the most relevant entities and relational dependencies. By using a relational model, evidence can be leveraged across entity and relation instances. By performing joint inference at query time, NLP pipeline errors are minimized, and more adaptive and discriminative models that meet the specific knowledge discovery needs of the user can be developed.

Our goal is to develop user-driven relational models of entities and their relational dependencies, and a search system based on these models that allow users to search for known entities and relations, discover new relations from known entities, and discover new entities from known relations. Preliminary qualitative and quantitative evaluations demonstrate the efficacy and potential of the proposed relational modeling approach.

Categories and Subject Descriptors

H.3.1 [**Content Analysis and Indexing**]: *Indexing methods*, *Linguistic processing*. H.3.3 [**Information Search and Retrieval**]: *Search process*: *Query formulation*, *Retrieval model.*

Keywords

Information Retrieval, Information Extraction, Relational Models, Knowledge Discovery.

SIRIR'12, 2012, Portland, OR, US.

1. INTRODUCTION

Information retrieval (IR) systems adapt to user needs by retrieving a set of documents that is *relevant* to an *adhoc* natural language query expressing an *information need*. The results are not precise and do not capture relational information, i.e., users seeking to understand how entities are related are forced to scan each document, extract relevant pieces of information, and assemble the extracted findings before they can solve their problem. Recall can be improved by adjusting query terms or scanning additional documents in the retrieved set.

In contrast to adhoc IR, information extraction (IE) systems process a collection of documents offline using extraction models (extractors) to identify precise named entities and relations. The most successful extractors are tailored to specific domains, or are limited to a set of general entities and relations involving people, locations, or organizations. Recall is fixed a priori by model thresholds.

Extractors use a variety of techniques ranging from *knowledge-based* encoding using hand-crafted rules and lexicons to *supervised learning* methods using hand-labeled training data. Knowledge-based extractors do not tend to generalize well to previously unseen examples, are labor intensive to create, and are not scalable as identification of new entities and relations require creation of new pattern matching rules or lexicon entries. Supervised methods [1,2] use domain-independent machine learning methods to automatically learn an extractor from a set of domain-specific training data. Supervised learning methods tend to work best for specific domains where training data is plentiful or for relatively basic extraction patterns.

In each case, extractors are created in *advance*, and new extractors must be created to meet new information extraction needs. Defining extractors in advance requires you to essentially know what information you are looking for before you can extract it. For example, let's say you are interested in identifying people who are involved with *financing terrorist activities*. Knowledge-based methods would necessitate the definition of new extraction rules and lexicon vocabulary. Supervised learning methods would require the assemblage of training data to learn a model to recognize this entity relation, provided such training data is available. Attempting to continue this process for all possible extractions at the appropriate level of granularity and precision/recall is clearly intractable.

Semi-supervised approaches address the problem of limited training data. The basic idea is to supply a small amount of

labeled training data to bootstrap learning of an extractor, use this extractor to identify additional examples of highly probable extraction patterns, use these examples as an additional source of training data, and repeat the process until some termination criteria is met. A significant issue with this approach is the problem of drift. Without manual intervention, these models eventually drift and generate false positives. Other issues include the relative simplicity of the relational patterns extracted, and the need to define an extractor for bootstrapping in advance.

A fundamental phenomenon of natural language is the variability of semantic expression where the same meaning can be expressed by, or inferred from, different texts. This variability can make traditional lexical and syntactic based information extractors relatively brittle for a broad range of entity relations. A more general approach is to represent relations by word dependencies. Bunescu and Mooney [11] observed that the information required to assert a relation between two named entities in the same sentence is typically captured by the shortest path between the two entities in the dependency graph.

A dual representation of semantic relations can be used to identify relations from known entities, and identify entities from known relations. Bollegala and Matsuo [15] proposed such a dual representation and an unsupervised sequential co-clustering algorithm that extracts relations from unlabeled data. Relations are represented *extensionally* by the sets of entities involved with that relation (Google, Youtube; Microsoft, Powerset), and *intentionally* by the properties or words of that relation (X *is acquired by* Y, or Y *purchased* X). We use this dual representation to facilitate information retrieval

Clearly, there is a critical need for knowledge discovery tools to facilitate search and extraction of adhoc fine-grained entity-relations that are specific to an individual's information need. To meet this need, we present a ranked retrieval and extraction framework for collectively learning and integrating evidence of entities and relational dependencies to predict at query time, a ranking of sentences containing the most relevant entities and relational dependencies. With this "*Everything is Miscellaneous*" [16] approach, user-defined semantic retrieval needs are defined at query time, bypassing the computationally intractable bottom up approach of most existing methods.

First we present an information seeking scenario to illustrate our approach. This is followed by our ranked retrieval architecture, relational methods for modeling of *entity-relations*, NLP methods for processing sentences and queries, relational indexing methods, retrieval and extraction models, results and evaluation, and prior work.

2. Information Seeking Scenario

Our proposed framework is modeled to support interactive knowledge discovery based on a dual representation of entities being defined by their relations, and relations being defined by their participating entities. In this illustrative scenario, we are in the role of an analyst who is interested in learning about terrorist movements. We know of a terrorist *Hambali* who has moved to *Malaysia*, so we start with the topic query "*Hambali moves to Malaysia*." The system extracts the following dependency relation from the query: *[hambali-1, moves-2, to-3, malaysia-4] (entity1, relational dependency sequence …, entity2)* and using a retrieval model that integrates evidence of *entities* and *relations,* retrieves the following sentence (Table 1) from a Web document.

Born and educated in Indonesia, Hambali moved to Malaysia in the early 1980s to find work. *(indonesia; in_born_move_to; Malaysia)* *(hambali; move_to; malaysia)* *(indonesia; in_born_move; hambali)*

Table 1. Entity-Relation Search Result for query: *Hambali move to Malaysia (entity; relation; entity)*

The relations extracted from the sentence provide a *relational lattice* linking *Indonesia*, *Malaysia*, and *Hambali*. From analyzing the results of the query, the analyst may be interested in identifying other *entities* participating in this *move to* relation. In this case, the analyst searches with a retrieval model using evidence of the *relation* (with any compatible *entity)* and retrieves the sentences (and relation extractions) shown in Table 2.

After returning briefly to Pakistan, he moved his family to Qatar at the suggestion of the former minister of Islamic affairs of Qatar, Sheikh Abdallah bin Khalid bin Hamad al Thani. *(pakistan; to_return_move_suggest_famili_to; qatar)*
KSM then accepted Bin Ladin's standing invitation to move to Kandahar and work directly with al Qaeda. *(ksm; accept_move_to; kandahar)*
In Iran, KSM rejoined his family and arranged to move them to Karachi; he claims to have relocated by January 1997 *(iran; in_rejoin_arrang_move_to; Karachi)* *(ksm; rejoin_arrang_move_to; Karachi)*
He is thought to have moved to Pakistan when the Taliban fell, and he may have gone to Yemen in recent months. *(pakistan; to_move_fell; taliban*

Table 2. Relation Search Result for query: *Hambali move to Malaysia (entity; relation; entity)*

The analyst can now identify new *entities* participating in some form of *move to* relation. From these results, other *relations* for one or more *entities* or any combination of *entity*, *relation*, or *entity-relation* can be explored by adjusting the query.

3. Ranked Retrieval Architecture

Figure 1 illustrates the ranked retrieval process used to support this information seeking behavior. The architecture supports the following process:

1. The user presents a natural language query.
2. The NLP engine parses the query, extracts candidate entities, relations, and textual context.
3. A relational query model is generated from the evidence the NLP engine was able to extract from the query.
4. The relation query model is used to rank sentences from the dimensional index.
5. The user can provide relevance feedback to the system.

The same NLP process is used for parsing document sentences and queries.

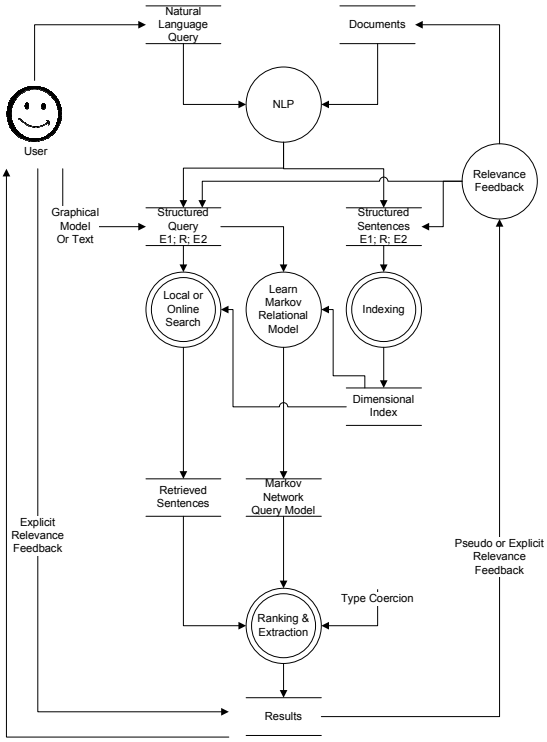

Figure 1. Ranked Retrieval Architecture

4. Relational Modeling of Entity-Relations

Entities, *relations*, and *word context* are related in complex ways and their classifications are *relationally* interdependent. We refer to the relationship between *entities*, *relations*, and *context* as an *entity-relation*. *Entities* can be defined by the *relations* they participate in, *relations* can be defined by the set of *entities* participating in the *relation*, and instances of *entity-relations* can be inferred from *entities, relations*, or word *context*. This suggests a relational model based on a multievidentiary lattice of relational dependencies.

Relational models have been shown to improve accuracy in applications with relational dependencies [17,18,19,26]. As shown in Figure 2, our proposed relational model provides a template for collectively capturing the relational dependencies of *entities*, *relations* (relational dependency between entities), and word *context* across multiple sentence-level instantiations.

An instantiation of this schema is represented as a Markov Network [20] extended for the relational setting [17] that includes all instances of *entities*, *relations*, and word *context* in a collection of documents that is compatible with a user query. The network captures the interactions between all related instances by allowing us to represent correlations between their attributes. From the model we can infer the conditional probability of a sentence generating the entities in an adhoc query from the likelihood of that sentence's relational dependencies and word context across all known instances of that entity type. Likewise, we can determine the likelihood of a relation between entities from the likelihood of those entities participating in that relation.

The resulting network captures the joint distribution of *word context*, *entities*, and *relations* given the collective evidence of related instantiations. For example, given a query with a full specification of an *entity-relation*, e.g., {*KSM; meeting with; Osama bin Laden}*, the network can be used to determine the likelihood of any sentence in the collection being relevant to that entity-relation. Given a query with a partial specification of an *entity-relation*, e.g., {*KSM meeting with <person>}*, the network can be used to discover the likelihood of any instance of a *<person>* entity being part the relation *meeting with* and the entity *KSM*. The framework can also be used to discover relations between individuals, e.g., {*KSM; bin Laden}*, or given a relation, the most likely entities participating in that relation, e.g., {*<person> running training camp <location>}*.

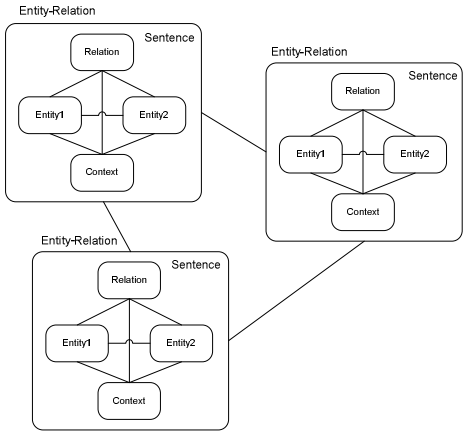

Figure 2. *Entity-Relation* Relational Model

An example instance of the *relational model* is shown in Fig. 2.

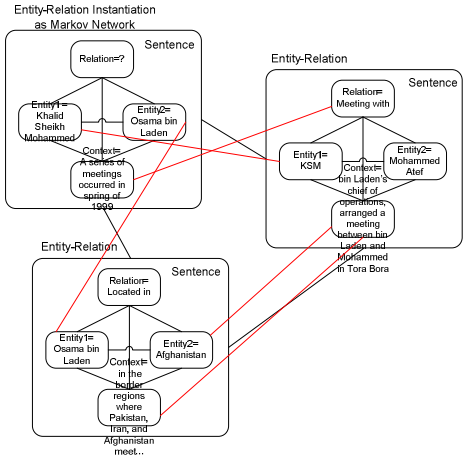

Figure 3. Instantiation of *Entity-Relation* Relational Model

5. Relational Indexing

To enable efficient search and extraction, inverted indexes are constructed for words, entities, and relations using a data warehousing style dimensional data model [21,22]. Each index is cross-indexed, i.e., *words* are related to *entities* and *relations*, and *entities* are related to *relations*. The grain of each index is the individual word, i.e., the grain of the word index is obviously a word, the grain of the entity index is each word participating in an entity instance, and the grain of the relation index is each word participating in a relation instance. This facilitates efficient vector-space cross-product SQL queries for aggregating query-time statistics. Each word includes parsing information and can be aggregated by phrase, entity, relation, sentence, paragraph, and document.

6. Sentence Processing

NLP of queries and sentences for indexing is listed below. Processing of a sample sentence is provided in Fig. 3:

1. The OpenNLP toolkit [23] is used to tokenize, tag, chunk, and identify named entities (person, organization, location, time, date, percent, money).
2. A dependency graph is created from the list of dependencies generated by the Stanford Parser [24].
3. Candidate entity pair relations are identified from each distinct pair of noun chunks, proper noun chunks, and named entities provided each pair contains at least one proper noun or named entity.
4. A dependency relation is extracted for each candidate entity pair by identifying the shortest path between the candidate entities in the dependency graph using Dijkstra's algorithm.

Sentence/Query: Sayad establish training camp in Pakistan

Sentence parse (term, POS, noun phrase, proper noun phrase):
Sayad NNP B-NP PNP
establish VB B-VP O
training NN B-NP O
camp NN I-NP O
in IN B-PP O
Pakistan NNP B-NP PNP

Entities: Location: 5:6 Pakistan;

Dependency relations: (entity1, dependency sequence, entity2)
[sayad-1, establish-2, camp-4, in-5, pakistan-6]

Figure 4. Sentence NLP

7. Retrieval Model

As described in section 4, our model is based on a Markov network. A Makov network is a model for the joint distribution of a set of variables $X = (X^1, X^2, \ldots, X^n)$. It is composed of an undirected graph G and a set of potential functions ϕ_k. The graph has a node for each variable, and the model has a potential function for each clique in the graph. A potential function is a non-negative real-valued function of the state of the corresponding clique. The joint distribution is given by:

$$P(X = x) = \prod_k \phi_k(x_{\{k\}}) \quad (1)$$

Where x_k is the state of the k^{th} clique, i.e., the state of the variables that appear in that clique. Z is a normalization constant known as the partition function:

$$Z = \sum_x \prod_k \phi_k(x_{\{k\}}) \quad (2)$$

For convenience, Markov networks are often represented by log-linear models where each clique potential is replaced by an exponentiated weighted sum of features of the state:

$$P(X = x) = \frac{1}{Z} \exp(\sum_k w_j f_j(x)) \quad (3)$$

A feature may be *any* real-valued function of the state. We are free to specify a factorization of the graph into a set of features, i.e., functions representing the affinity or likelihood of the state of the clique. We can take advantage of this by specifying cliques for local and global (relational) distributions for our component entity, relation, and word context models.

Since we are interested in ranking, we avoid the computational cost of calculating the partition function by normalizing each feature function prior to inclusion in the weighted sum.

7.1 Entity Feature Function

As shown in equation (4), the probability of an entity being generated for a given sentence is estimated using a Jelinek-Mercer style weighting of the *presence* of the entity, and the likelihood of the sentence *entity-word* distribution from equation (5).

$$p(e_j \mid s_k) = \lambda * presence(e_j) + (1-\lambda) * p_d(e_j \mid s_k) \quad (4)$$

The *presence()* function represents the probability of matching an entity in the relational model generated from the query. For entity instances, this is a term match. For entity types, this is an entity type match. If no entity is specified, it is treated as a wildcard. The probability of a match can be efficiently calculated as the cross product of the normalized IDF of the candidate and target entities from the dimensional index. The sentence *entity-word* distribution is estimated from the entity-word co-occurrence (Eq. 5).

$$p_d(e_j \mid s_k) \approx \sum_{i=1}^{|S|} p(w_i \mid e_j) p(e_j) \quad (5)$$

7.2 Relation Feature Model

The likelihood of a *relational dependency* is calculated as the cross product of the normalized inverse relational dependency frequency (NIRDF) of the candidate relation terms and the terms of the target relation extracted from the query. The NIRDF is calculated over terms within relational dependency sequences to provide a measure of relational term specificity.

7.3 Term Feature Model

We generate term context models for sentences and documents. The likelihood of a sentence or a document generating the terms in the query is approximated using a normalized version of Robertson's BM25 [25] similarity coefficient.

7.4 Aggregate Network Models

Aggregate network models are generated by instantiating a network of any combination of feature models parameterized by the user supplied query (equation 3).

8. Evaluation

A prototype of the relational modeling framework was developed as a distributed Java/MySQL/Tomcat Web application and deployed on the Amazon Web Services cloud computing infrastructure. We performed our evaluation using the ACE 2005 newspaper data set [14] for each of the models listed in table 3. Each model is described in section 6.

Model (Abbreviation)	Feature functions included in model
Aggregate (A)	Entity + sent term + document term
Term (T)	Sent term + document term
Entity (E)	Entity
Entity-Relation (E-R)	Entity + relation
Relation (R)	Relation

Table 3. Models Evaluated

Table 4 lists the queries used for the evaluation. The queries were selected based on the topics available in the data set, and as demonstrated by the NLP parse of each query, the varying amount of entity-relation evidence for analysis.

	Query	NLP Parse of Query
1	Journalist killed in Baghdad.	[journalist; killed in, Baghdad]
2	Fighting in Fallujah	[*; fighting in; Fallujah]
3	Kurdish, political leaders	[Kurdish; political leaders;*]
4	China, relations with	[China; relations with;*]
5	go to, travel to, fly to, went to, get out of	[*;go to, travel to, fly to, went to, get out of;*]
6	Pearl was murdered by terrorists in Pakistan	[Pearl; was murdered by terrorists in Pakistan
7	Indonesia meeting with Putin	[Indonesia; meeting with; Putin]
8	CIA has technology	[CIA; has technology;*]
9	Khalid Shaikh Mohammed capture in Pakistan	[*;*;Pakistan]
10	Jack Welch seeks details on estranged wife	[Jack Welch;*;*]

Table 4. Evaluation Queries

As shown in table 5, a qualitative evaluation was performed for each query and each retrieval model's result set to gain insight into the effectiveness of each model given the available evidence extracted for each query.

Query 1:	Journalist killed in Baghdad
Entities:	Location: 3:4 Baghdad;
Extracted query rel.	[journalist-1, killed-2, in-3, baghdad-4]
Sentence result + [*extracted relation*]:	Journalists killed in the line of duty in Baghdad -- how neutral are reporters in a war supposed to be? *[journalist; kill_in_line_of_duti_in; Baghdad]*

Table 5. Sample Query/Model Evaluation

The E-R model capturing compatible entities and relations significantly outperformed models using entities, terms, or relations alone, or entities and terms in combination. A significant part of the failure of term models is due to the relatively low IDF of terms that are typically part of a relation. Similarly, entity and aggregate entity-term models barely outperform basic term models without being integrated with relational terms. Of special note is the high precision of the relation model (R-prec) for identifying relations with compatible entities.

Quantitative results are shown in Table 6. Relevance judgments were made from a pool of the top 20 sentences retrieved from all models for each query. Relevance judgments were based on the intended meaning of entities or relation expressed in each query. Relevance for the R-prec (relational precision) was determined *only* from the intended meaning of the relation expressed in the query. The idea here is to gain insight into the generalizability of relational dependencies.

9. PRIOR WORK

Brin [3] proposed *DIPRE – Dual Iterative Pattern Relation Expansion*; a semi-supervised learning technique that exploits the duality between sets of *patterns* and *relations* to grow the target relation starting from a small sample. Starting with a small seed set of *(author, title)* pairs to define a book relation, the web is searched to find all occurrences of those books. From these occurrences, regular expression patterns are created from the *prefix, author, middle, title* and *suffix* of retrieved book citations. The book *patterns* can now be used to search the web to find occurrences of new books. From these new books, all of their occurrences can be found, and from those patterns, more patterns can be generated and so forth. The process continues, and a list of books and patterns are generated for finding them.

Agichtein, et al. [4] expanded on *DIPRE* with the *Snowball System* for extracting relations. They used a semi-supervised learning algorithm similar to Brin's, and proposed two alternative methods for representing the textual contexts around relation extractions: unordered keywords and ordered keywords. By combining the results from both textual contexts, they were able to demonstrate significantly improved performance. What is notable from this work is the effectiveness and flexibility of using keywords without any significant structure to represent relations.

Query	A	E	T	ER	R	R-prec	n
1	0.63	0.50	0.42	0.78	0.78	1.00	13
2	0.00	0.50	0.42	1.00	1.00	1.00	1
3	0.00	0.00	0.00	0.33	0.80	1.00	2
4	0.00	0.00	0.00	0.86	0.60	1.00	4
5	0.00	0.00	0.00	0.92	0.92	0.86	6
6	0.00	0.00	0.00	1.00	1.00	0.30	1
7	1.00	1.00	1.00	1.00	0.67	0.80	1
8	0.93	0.93	0.93	0.62	0.15	0.14	8
9	0.80	0.80	0.80	0.50	0.00	0.71	2
10	0.88	0.88	0.80	0.63	0.00	1.00	8
Avg.	0.42	0.46	0.44	0.76	0.59	0.78	
Median	0.31	0.50	0.42	0.82	0.72	0.93	

Table 6. Results: A, E, T, E-R Models (F-Score), R-precision

Mitchell, et al. [5] proposed a *macro-reading* approach that does not make any attempt to extract all information within a document (*micro-reading*) and instead relies on the availability of the large amount of redundant information that is available on the Web. They ignore complicated sentences and statistically combine evidence. To constrain the learning problem of reading free-form text, they formulate the *macro-reading* problem as a task of populating an ontology that is given as input that defines the categories and relations of interest. The system can focus only on a subset of text that is *on-topic* with respect to the ontology and its meta-properties. Semi-supervised learning (co-training) from a handful of labeled examples is used to bootstrap learning of extraction patterns. This semi-supervised learning approach was extended by Carlson, et al. [6] to extract entities and relations (e.g., plays Sport (athlete, sport)) from web pages starting with a handful of labeled training examples of each category or relation, plus hundreds of millions of unlabeled web documents. The approach is limited to domains with large amounts of redundant data and basic relational patterns. A small amount human intervention is necessary to prevent drift of learned relational patterns.

Traditional IE methods learn lexical (word) models of individual relations from hand-labeled examples of sentences that express these relations. Lexical features are relation specific, but when

using the Web as a corpus or any newly identified collection of data, relations are not known in advance. Schubert [7] proposed an information extraction model that first learns a general model of how relations are expressed in a particular language, and then used this model as the basis of a relation-independent extractor whose sole input is a corpus and whose output is a set of extracted tuples that are instances of a potentially unbounded set of relations. Etzioni, et. al [8] used a similar approach in their Open Information Extraction model. Like Schubert, their approach is based on the assumption that you can define a general model for English. They claim 95% coverage of relations. To correct for uninformative and incoherent extractions, Afader, Soderlan, and Etzioni [9] added syntactic and lexical constraints to their Open IE approach in the Reverb system.

Sekine [10] proposed an unsupervised method to discover *paraphrases* from a large untagged corpus without requiring a seed phrase or other clue. The procedure starts with the identification of two named entities and uses the following criteria: If two phrases can be used to express the same relationship within an IE scenario, these two phrases are paraphrases.

In contrast to phrase structure grammars, structure in dependency grammars is determined by the *relation* between a word (a head) and its dependents [12,13]. This has lead to defining entity relations by the dependency structure between entities. In experiments extracting top-level relations from the ACE [14] newspaper corpus Bunescu and Mooney [11] demonstrated that capturing the dependency structure between two entities outperformed several other extraction methods.

Bollegala and Matsuo's [15] dual representation of semantic relations uses individual words and part-of-speech for features. They claim to outperform state of the art Open IE systems in terms of precision and recall. Co-clustering and sub-sequence mining is used to define semantic relation classes and control the explosion of candidate extractions.

10. Conclusion

We have presented a ranked retrieval and extraction framework for collectively integrating evidence of entities and relational dependencies to predict at query time, a ranking of sentences containing the most relevant entities and relational dependencies. In doing so, we have introduced a novel user-driven approach integrating *entity-relation* retrieval and extraction. Preliminary results demonstrate the efficacy of our approach using relatively basic models. Future work includes evaluation on larger and more diverse data sets, online learning of models parameters via user relevance feedback, user annotation of entity and relation types, additional research into ranked relational models, and deployment in a high-performance computing environment.

11. ACKNOWLEDGMENTS

This material is based on past research sponsored by the Air Force Research Laboratory and Air Force Office of Science and Research Visiting Faculty Research and Summer Faculty Fellowship Programs (2010-2011) agreement number (13.20.02.B4488), and current research being sponsored by the Air Force Research Laboratory under agreement number (FA8750-12-1-0031). The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

12. REFERENCES

[1] S. Soderland. Learning information extraction rules for semi-structured and free text. Machine Learning 34, 1–3, 1999, 233–272.

[2] E. Riloff. Automatically constructing extraction patterns from untagged text. In Proceedings of the 13th National Conference on Artificial Intelligence, 1996, 1044–1049.

[3] S. Brin, Extracting Relations from the World Wide Web. Proceedings of the Workshop at the 6th International Conference on Extending Database Technology, Valencia, Spain, 1998.

[4] E. Agichtein, and L. Gravano. Snowball: Extracting relations from large plain-text collections. In Proceedings of the 5th ACM International Conference on Digital Libraries, 2000.

[5] T. Mitchell, J. Betteridge, A. Carlson, E. Hruschka Jr., and R. Wang. Populating the Semantic Web by Macro-Reading Internet Text. Invited paper. Proceedings of ISWC, 2009.

[6] A. Carlson, J. Betteridge, R. Wang, E. Hruschka Jr., and T. Mitchell. Coupled Semi-Supervised Learning for Information Extraction. Proceedings of WSDM, 2010, New York, NY.

[7] L. Schubert. Can we derive general world knowledge from texts? In Proceedings of the Human Language Technology Conference, 2002.

[8] O. Etzioni, M. Banko, S. Soderland, and D. Weld. Open Information Extraction from the Web. Communications of the ACM, Dec. 2008, Vol. 51, #12.

[9] A. Fader, S. Soderland, and O. Etzioni. Identifying Relations for Open Information Extraction. EMNLP '1, 2011.

[10] S. Sekine. Automatic Paraphrase Discovery based on Context and Keywords between NE Pairs, 2005.

[11] R. Bunescu and R. Mooney. Shortest Path Dependency Kernel for Relation Extraction. Proceedings of the Conference on Empirical Methods in Natural Language Processing, Vancouver, B.C., pp. 724-731, October, 2005.

[12] Lucien Tesnière. Éléments de syntaxe structurale, Klincksieck, Paris 1988. Preface by Jean Fourquet, professor at Sorbonne. Revised and corrected second edition, 1988.

[13] R. Snow, D. Jurafsky, and A. Ng. Learning syntactic patterns for automatic hypernym discovery. NIPS 17, 2005.

[14] Christopher Walker, et al. ACE 2005 Multilingual Training Corpus, Linguistic Data Consortium, Philadelphia, 2006.

[15] Danushka Bollegala, Yutaka Matsuo, Mitsuru Ishizuka. Relational Duality: Unsupervised Extraction of Semantic Relations between Entities on the Web. WWW 2010, April 26-30, 2010. Raleigh, NC.

[16] David Weinberger. Everything is Miscellaneous. Holt, 1st Ed., 2008.

[17] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. Proceedings on Uncertainty in AI, 2002.

[18] J. Neville and D. Jensen. Iterative classification in relational data. In Proceedings of the AAAI-2000 Workshop on Learning Statistical Models from Relational Data, 2000.

[19] L. Getoor, N. Friedman, D. Koller, and B. Taskar. Learning probabilistic models of relational structure.ICML, 2001.

[20] J. Pearl. Probabilistic Reasoning in Intelligent Systems. Morgan Kaufmann, San Francisco, 1988.

[21] J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M Venckatrao, F Pells. Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals. Data Mining and Knowledge Discovery, Vol. 1, Issue 1, 1997.

[22] J. Urbain, O. Frieder, and N. Goharian. Passage relevance models for genomics search, BMC Bioinformatics 2009, 10 (Suppl 3):S3.

[23] Open NLP, acc, Jan. 28, 2012. http://incubator.apache.org/opennlp/

[24] M. de Marneffe, B. MacCartney and C. Manning. Generating Typed Dependency Parses from Phrase Structure Parses. In LREC 2006.

[25] S. Robertson and S. Walker. Okapi/Keenbow at TREC-8,"NIST Special Publication 500-246, 2000.

[26] D. Metzler, B. Croft. A Markov Random Model for Term Dependencies. SIGIR'05, 2005.