

It has been a long term dream of AI to develop systems that can emulate human levels of language understanding and reasoning, for example to answer questions or to construct summaries. Recent foundational, methodological and technological developments in Knowledge Representation (e.g. ontologies, knowledge bases incorporating various forms of incompleteness or uncertainty), Reasoning (e.g. data fusion-integration, argumentation, etc.), Natural Language Processing (such as information extraction, relation detection) and formal pragmatics (user models, intentions, etc.) make it possible to foresee the elaboration of much more accurate, cooperative and robust systems dedicated to understanding, learning and answering questions from textual data, operating either on open or closed domains. The time is right to start placing the pieces from all these different areas to develop a unified system for Learning by Reading and Automated Question Answering.

The goal of this workshop is to bring together researchers from different backgrounds (AI, NLP, linguistics, HLT and pragmatics) to explore possibilities of integrating the different techniques for building a system for Learning by Reading. The workshop will be focused on models for intelligently analyzing data and constructing data repositories.

This workshop starts with two papers dedicated to question answering: the first one, by Bourse et al. deals with a model of explanation structure appropriate for generating explanations in question-answering following standard schemas as found in technical documents. The second paper, by Barker et al. shows improvement strategies for conjunctive queries, considering that errors in individual query conjuncts multiply in conjunctive ones, often causing recall to drop dramatically as queries add terms.

The next three papers are dedicated to learning from texts. The first paper, by Garcia et al. the authors propose to select four main types of linguistic features and systematically evaluate their performance in semantic Relation Extraction. The second paper, Popescu et al. present an unsupervised and language independent method of acquiring quasi-similar pairs which capture world knowledge. One of the aims is to improve the quality of textual entailment. The last paper, by Radinsky et al., present a new methodology for modeling and predicting future events through machine learning and data mining techniques from textual data.

The next group of papers, by Doppa et al. and Sorower et al. present two methods for learning rules from incomplete examples, via Observation models and a Probabilistic Mention model.

This workshop ends by four papers that describe ongoing work of much interest: M. Glass et al.: a bootstrapping method to extract relation information from parallel corpora, Kim et al., present the use of information sources for resolving the ambiguities captured in a packed representations, while Popescu

et al. introduce a method for Textual Entailment strategies Using Chain Clarifying Relationships. Finally, Samadi et al. present an approach for evaluating Correctness of Propositions Using the Web.

This workshop is a series of several that merges here the FAM and the KRAQ series. The number of papers submitted confirms the motivations for merging visions that complement each other.

This workshop was co-organized by Rutu Mulkar-Mehta and Patrick Saint-Dizier (co-chairs) and Eduard Hovy, Marie-Francine Moens, Bernardo Magnini, and Chris Welty. We thank the reviewers who contributed to the success of this workshop.

Keyword Index

Automatic Evaluation of learned Knowledge	41
Bootstrapped learning	8
Bootstrapping Learning	20
Bridging knowledge gaps in NLP	1
communication goals	2
Cooperativity	2
Corpus Construction	20
discourse semantics for QA	2
domain knowledge	45
information extraction	14
Information Extraction	41
Knowledge extraction from text	20, 41
Learning Causality from text	35
Learning temporal sequences from text	35
machine reading system	21
markov logic networks	45
Mechanical Turk	20
mention model	45
NLU	21
Observation models	8
ontologies	14
Ontology learning, population, or expansion	35
packed representation	21
Query analysis	1
Question answering systems	1
Relation Extraction	20
Rule learning from incomplete examples	8
semantic relation extraction	14
Social Roles	29
supervised	25
systematic bias	45

Text Entailment	29
Textual Entailment	25
unsupervised corpus learning methodology	29
wikipedia	14
Word Sense Disambiguation	25

Program Committee

James Allen	Florida Institute for Human
Machine Cognition	
Ken Barker	University of Texas at Austin
Danny Bobrow	PARC
Johan Bos	University of Groningen
Eric Brown	IBM TJ Watson Research Center
Nate Chambers	Stanford University
Peter Clark	The Boeing Company
Ido Dagan	Bar Ilan University
Oren Etzioni	Turing Center, Univ. Washington
Marjorie Freedman	BBN
Ralph Grishman	New York University
Sanda Harabagiu	University of Texas at Dallas
Jerry Hobbs	Information Sciences Institute
Eduard Hovy	ISI / University of Southern California
Dietrich Klakow	Saarland University
Zornitsa Kozareva	Information Sciences Institute
Chin-Yew Lin	Microsoft Research Asia
Bernardo Magnini	FBK-irst
Mausam Mausam	University of Washington
Marie-Francine Moens	Katholieke Universiteit Leuven
Dan Moldovan	University of Texas Dallas
Rutu Mukherjee	Information Sciences Institute
Tim Oates	University of Maryland Baltimore County
Anselmo Peñas	UNED
Bruce Porter	University of Texas at Austin
Silvia Quarteroni	Politecnico di Milano
Patrick Saint-Dizier	IRIT - CNRS
Len Schubert	University of Rochester
Luciano Serafini	FBK-IRST
Christopher Welty	IBM Research
Michael Witbrock	Cycorp

Additional Reviewers

Blanco, Eduardo

Christensen, Janara

Fader, Anthony

Soderland, Stephen

Table of Contents

An analysis of explanation structures for answering complex questions	1
<i>Sarah Bourse and Saint-Dizier Patrick</i>	
An Exploration of the Linguistic Knowledge for Semantic Relation Extraction in Spanish .	7
<i>Marcos Garcia and Pablo Gamallo</i>	
Learning Quasi-Similar Pairs	13
<i>Octavian Popescu, Emanuele Pianta and Bernardo Magnini</i>	
Bootstrapping Relation Extraction Using a Parallel Corpus	19
<i>Michael Glass and Ken Barker</i>	
Improving Recall on Conjunctive Queries against Text Documents using Query-Driven Hypothesis Generation	20
<i>Ken Barker, James Fan and Chris Welty</i>	
The packed representation for delaying ambiguity resolution and the information sources for resolving the ambiguities captured in the packed representation	26
<i>Doo Soon Kim, Ken Barker and Bruce Porter</i>	
Learning Rules from Incomplete Examples via Observation Models	30
<i>Janardhan Rao Doppa, Mohammad Nasresfahani, Mohammad S. Sorower, Jed Irvine, Thomas G. Dietterich, Xiaoli Fern and Prasad Tadepalli</i>	
Textual Entailment Using Chain Clarifying Relationships	36
<i>Octavian Popescu, Elena Cabrio and Bernardo Magnini</i>	
Mention Model for Learning Rules from Incomplete Examples	40
<i>Mohammad Sorower, Thomas Dietterich, Janardhan Doppa, Prasad Tadepalli and Xiaoli Fern</i>	
Evaluating Correctness of Propositions Using the Web	46
<i>Mehdi Samadi, Manuela Veloso and Manuel Blum</i>	
Learning causality from textual data	50
<i>Kira Radinsky, Sagie Davidovich and Shaul Markovitch</i>	

An analysis of explanation structures for answering complex questions

Sarah Bourse, Patrick Saint-Dizier

IRIT - CNRS,

118 route de Narbonne, 31062 Toulouse Cedex, France.

sarah.bourse@gmail.com, stdizier@irit.fr

Abstract

In this paper, we introduce a study of explanations as they appear in technical documents, in particular procedures where they are crucial. Identifying how explanations are structured in such types of productions is essential for example to be able to produce explanations in question-answering systems. Our study is based on an extensive corpus analysis. We introduce the notion of explanation function, serving communication goals, and then motivate and illustrate the introduction of explanation schemes which are the language realizations of the functions. We give some details about linguistic patterns that allow for the identification of these various schemes.

1 Introduction and Context

The main goal of our project is to elaborate relatively simple and direct explanation schemes, serving specific communicative goals, in order to offer more cooperative responses in question-answering systems. Indeed, a number of questions cannot be answered directly and therefore require some forms of explanations so that the user understands the details and the reasons of the information he gets, for example when the answer is produced via the relaxation of some constraints in the question. Our approach consists in identifying simple and operational explanation schemes in texts where style is direct and efficient, and then to transpose these results to question-answering. For that purpose, we explore in this paper quite a large diversity of procedural texts. Procedural texts are indeed very rich in explanation forms which are of much interest for question-answering systems. We consider here a large diversity of types of texts from gardening, DIY, health care, to didactics.

Explanation and its relations to language and linguistics is a relatively new but vast area of investigation in computational linguistic circles. From works in RST theory (Mann and Thomson, 1988), let us cite efforts e.g. by (McKeown, 1985) and by a number of protagonists of the natural language community in the early 1990. Let us cite the It involves the taking into account of a large number of linguistic aspects, from syntax to pragmatics, but also typography. At the moment, explanation is essentially developed in various sectors

where didactics is involved, e.g. writing recommendations for producing essays or technical documents or, in interactive environments, systems such as helpdesks. Two decades ago, explanations were used to produce natural language outputs for experts systems, often from predefined templates. The goal was to justify why a certain proof was carried out and why a certain solution was proposed as a result of a query. In the same range of ideas, and more generally, natural language generation planning made some use of explanation structures (e.g. McKeown, 1985). Our perspective in this analysis is to be able to recognize and produce explanations in question-answering situations where they are necessary (e.g. indirect response to a query). The work presented here is the first stage: analysing the structure of explanation from a language and communication point of view. An interesting contribution (Zuckerman et al., 2000) shows how argumentation (an aspect of explanation) can be organized within the framework of natural language generation.

Interesting principles have emerged from these works which have motivated the emergence of more foundational research. For example, in ergonomics and cognitive science, the ability for humans to integrate explanations about a task described in a document or on an electronic device (possibly via a guidance system) when they perform that task is investigated and measured in relation with the document properties (typography, pictures) (Lemari et al., 2008), (Bieger and Glock, 1984). In linguistics, a lot of efforts have been devoted to the definition and the recognition of discourse frames (Webber et al., 1990, 2004; Miltasaki et al., 2004; Saito et al., 2006) and the linguistic characterization of rhetorical relations (Mann and Thompson, 1988, 1992; Longacre, 1982), which are, for some of them, central to explanation (Rosner and Stede, 1992; Van der Linden, 1993). However, we now observe a proliferation of rhetorical relations with various subtleties, which, for some of them, turn out to be quite difficult to recognize from language marks since they involve quite a lot of pragmatic considerations and domain knowledge. Finally, explanation is a field which is investigated in pragmatics (e.g. cooperative principles, dialogue principles) (Pollock, 1974) and in philosophy (e.g. rationality and explanation, phenomenology of explanation, causality, etc.) (Keil and Wilson, 2000; Wright, 2004; Davidson, 1980).

Explanation is in general structured with the aim of reaching a goal. This goal may be practical (e.g. how to reach

a certain location) or more interpersonal or epistemic (e.g. convince someone to do something in a certain way, negotiate with someone while providing explanations about one's point of view). Explanations are in fact often associated with a kind of instructional style, explicit or implicit, which ranges from injunctive to advice-like forms. Procedures of various kinds (social recommendations, as well as do-it-yourself (DIY), maintenance procedures, health care advice, didactic texts) form an excellent source of corpus to observe how explanations are constructed, linguistically realized and what aims they target. Procedures are composed of a skeleton, composed of a goal (possibly subgoals) and sequences of instructions designed to reach this goal. This skeleton is paired with a very large number of forms of explanations meant to guide, support, help, etc. the user. This is very useful to build a corpus of explanations since the language is often simple and direct and often the basis of How-to? questions or advice questions. In procedures, style is often straightforward (DiEugenio and Webber, 1996), procedures being essentially oriented towards action: there must be little space for inferences. This kind of corpus is particularly well-adapted for our investigation: it covers quite a large proportion of situations, reproducible, e.g. in authoring tools, helpdesks or in natural language generation systems.

Within the context of procedural texts, our main goal is to identify a number of prototypical, widely used, explanation schemes, their linguistic basis (e.g. prototypical language marks or constructs, planning issues), and to categorize their communicative goals. We aim at identifying the language and pragmatic means, given a certain aim, which are used to help, support, motivate or convince a reader. Then, we can apply these schemes to produce explanations in question-answering scenarios.

For that purpose, we consider a large variety of procedures, from large public ones (cooking, DIY), to professional ones (maintenance, health), with a large diversity of authors, target readers and application domains. We have a corpus of about 8000 procedures from 24 different domains out of which we have randomly extracted a development corpus of 1700 procedures dedicated to this study. The work was carried out for French and English; for the sake of readability, all examples are given here in English.

In this paper, we first present our corpus analysis strategy and methodology, and then introduce the notion of explanation function that specifies communicative goals of explanations. Explanation functions are abstract constructs which are realized in language via what we call explanation schemes. These notions applied to explanations and the way they are organized are new - to the best of our knowledge - and constitute our contribution, although they owe much to general considerations of rhetorical relation theory. An important feature of explanation is that schemes are transcategorical: they include syntactic and lexical semantics factors, as well as typographic (e.g. icons) and pragmatic factors. However, images, diagrams, and other multimedia devices are not considered here. Finally, under investigation is the notion of proto-relation for rhetorical relation with a wide scope such as elaboration, with an approach similar to thematic protoroles, as defined by D. Dowty.

2 Corpus Analysis and Annotation

In order to identify explanation functions and then in order to propose general explanation construction principles, we have first carried out an annotation of a portion of our corpus. This was realized by 3 Master students in linguistics on 74 different texts from our development corpus (about 88 pages), with the same training and annotation instructions.

Constructing a corpus in general is a very challenging task: it is first necessary to identify the parameters we want to measure, while keeping the others constant. Then, the scope of each parameter needs to be defined (which values they can take) and the distribution we want to have (the percentage of texts for each parameter value). The next step is the corpus construction, and its validation w.r.t. these parameters. Our parameters include taking into account the diversity of domains, authors, type of style, target audience and difficulty of procedure. The 74 selected texts allow us to make an indicative analysis that gives useful research directions and helps to confirm our working method. This small corpus of texts is composed as follows, with the following parameters and distribution rates among their values:

Domain: cooking: 7, gardening: 9, DIY: 12, health care: 8, didactics: 8, video game solutions: 5, social relations management: 9, beauty: 4, technology: 12.

Authors: teenagers: 15, professionals: 42, others: 17.

Target: large public: 47, students: 17, professionals: 10.

Style: poor: 25, standard: 39, high level: 10.

Difficulty of procedure: basic: 8, simple: 12, average: 24, difficult: 30.

The second step is the annotation. This is a difficult task: identifying and categorizing rhetorical relations is almost never straightforward. To help annotators, the task was realized from the outputs of our <TextCoop> system, that tags, with quite a good accuracy, instructions, goals, various discourse markers, arguments (previously studied and implemented), some manners, instruments, conditional expressions and simple causal forms (Delpech and Saint-Dizier, 2008). The tags have received a definition from previous works in discourse analysis, and have been made more precise and as unambiguous as possible considering the type of document we are dealing with, which are objective and explicit and which may follow authoring instructions. Two annotators have been trained, and after an adaptation period, annotators could do a homogeneous task. They annotated in parallel the same set of texts, and reached a relatively good consensus of 86%, measured by a Kappa test, this is very high, but it must be noted that procedural texts often follow authoring recommendations which facilitates their understanding. Over the 74 texts, 1127 structures related to explanation have been annotated, a ratio of 15 annotations per text. Distribution over texts ranges from 3 to 28 which often depend on the length of the text but also on the domain and the target audience. We kept in our analysis those annotations that occur at least 10 times over the whole set of texts, which corresponds to a frequency of about 1%, which sounds minimal to us to be really of interest. However, some non-frequent structures may be of interest because they introduce interesting views on explanation structures.

We basically consider the following tags, called hereafter Elementary Explanation Structures (EES), definitions are given here informally, and they follow from commonly admitted definitions that we have somewhat simplified considering the type of document used:

- elaboration and evaluation: elaboration (adds new information), illustration (provides related examples), reformulation (simply restates, does not add any information), result (specifies the outcome of an action), contrast (introduces a comparison by means of a major difference between two methods, objects or situations), analogy (form of comparison by image to help understanding), encouragement, hint, evaluation (explains a user what kind of result he should get or evaluates his performances).
- arguments: warning, advice (very few threats or rewards, where the author is involved). These often have the form of a causal expression, formally: do X 'because/otherwise' Z, with quite a large variety of causal marks (Fontan and Saint-Dizier, 2008).
- condition: It involves at least 2 structures: condition and then - else, and the assumption structure, which is an hypothetical statement. Condition is in general introduced by 'if' or 'when'.
- cause: It involves a statement and an ensuing situation as defined in (Talmy, 2001). We limit annotations to trans-sentential causal expressions, i.e. those operating over instructions.
- concession: statement associated with a concession statement (a general rule is given followed by an exception that could be admitted).
- goal expression: purpose, following (Talmy, 2001).
- frame: circumstance (of the realization of an action), and some propositional attitudes such as: commitment, authority.

Annotations can be embedded, leading to complex structures. In our corpus, annotations are in XML with some attributes, such as, e.g. the strength or weight of arguments, and meta-annotations such as the certainty of the annotator. A given structure may receive several annotations in case of ambiguity, overlap or multiple functions, this is expressed by the symbol /.

An example, in readable form, borrowed from didactics, is the following (most marks produced by our <TextCoop> system are omitted, only EES are given):

[*procedure* [*purpose* Writing a paper: [*advice* Read light sources, then thorough]]

[*assumption/circumstance* Assuming you've been given a topic,] [*circumstance* When you conduct research], move from light to thorough resources [*purpose* to make sure you're moving in the right direction].

Begin by doing searches on the Internet about your topic [*purpose* to familiarize yourself with the basic issues;]

[*temporal – sequence* then] move to more thorough research on the Academic Databases;

[*temporal – sequence* finally], probe the depths of the issue

by burying yourself in the library.

[*warning* Make sure that despite beginning on the Internet, you don't simply end there.

[*elaboration* A research paper using only Internet sources is a weak paper, [*consequence* which puts you at a disadvantage...]]

[*contrast* While the Internet should never be your only source of information, it would be ridiculous not to utilize its vast sources of information.]

[*advice* You should use the Internet to acquaint yourself with the topic more before you dig into more academic texts.]]

From these pages of annotated texts, we can induce more abstract regularities related to (1) the communicative goals and (2) the ways these goals are realized. For example, behind 'advice' or 'purpose' there is a communicative goal, designed respectively to help the user via a suggestion and to explain him the reasons of an action.

3 Explanation Functions

From our corpus investigations, and from our experience of how procedures are written and used by technicians or casual users, we can propose a first, global classification of quite a large number of 'conceptual' functions that realize the communicative goals mentioned above. These are called **explanation functions**. To carry out this task, our strategy was to identify and then categorize and structure the underlying communicative aims associated with our annotations or groups of annotations, producing a second level of annotations. Again, this is somewhat intuitive, but this was realized as a collective task with the aim of reaching a consensus. We present here the main categories and their organisation, which seems to be quite stable. To go deeper into more specialized explanation functions, we obviously need to carry out further investigations, testing and refinements.

Within an operational context in a broad sense, explanation functions can schematically be subdivided into two fields: the motivations of doing something ('**Why do action A?**') and the way to make something ('**How-to do A?**'). This view sets a more global analysis of action with the dichotomy intentions or motivations on the one hand and realization and its facets on the other, as can be found in Action Theory under a more philosophical view.

To avoid any confusion with EES names or with any existing terms, our explanation functions, which are abstract constructs, possibly complex, are prefixed with E-. Let us now introduce these functions. These remain so far abstract: their language realizations are accounted by schemes, given in the next section. Since we worked on French and English texts in parallel, for the sake of readability, examples in English are given here. Classification is stable over these two languages (and probably over a large number of languages) and linguistic realizations are often very similar.

The first subset, the motivations ('why do A?') functions, is composed of information providers: E-structure, E-information and E-arguments. The function **E-structure** enhances or makes more explicit the structure and the coherence of the text. It is in a large part composed of low-level goal and function expressions (*push the white button [to open*

the box]), indicating motivations and expected results. Goal expressions have a low scope and appear in general at the end of the statement, as confirmed by ergonomic studies. This function also includes several forms of low level structures, among which temporal structures realized by means of temporal connectors, punctuation or typography. The aim of the **E-information** function, which operates at the ideational level, is to enhance, reinforce, weaken or contradict the beliefs of the reader, as anticipated by the author of the text, by providing more specific information on some aspects of the action at stake (*Adding salt to your sauce is unnecessary because fish sauce is already salted*). This very general function may be subdivided into more accurate functions such as, e.g. **E-clarification** (*Poke the wire into the bottom of the flower [(where the stem was)] as far as you can without it coming out the other side*) or **E-precision** (*Hang the hanger in a dark area and wait for the flowers to dry. [A full drying process will take anywhere from 2 to 3 weeks.]*), still to be investigated.

The second subset of this group operates at the interpersonal level, and aims at motivating the user to realize the action at stake as required and as accurately as possible, via some precautions or recommendations. This subset is formed of the various types of arguments, as usually found in argumentation classifications, the main ones we have identified are: **E-warnings** (*Carefully plug in the mother card vertically otherwise you risk to damage its connectors*) and **E-advice** (*We recommend professional products for your leathers, they will offer a stronger protection while repairing some minor damages*), when there is no implication on the author's part, and **E-threats** (*You must confirm your connection within 30 minutes otherwise we will cancel it*) and **E-rewards** (*We suggest you to pay when you book your flight since a discount coupon will be offered to you for your next purchase*) otherwise. These are designed to justify the importance of an action and the necessity of doing it as required (warnings) or to indicate the optional character of an action (advice) and the benefits of doing it. Following several authors including (Walton et al., 2008), arguments are composed of two parts: the conclusion, which is the statement, and its support, which is the reason of the following, e.g. the advice or the warning. Besides the recognition of arguments, evaluating their illocutionary or persuasion strength is of much interest. This is realized in general via a series of marks, essentially adverbial, denoting various levels of intensity.

The second subset, the way to realize an action, i.e. 'How-to-do A?' functions, contains several families. The first one deals with functions related to controls on the user related to his actions, while the second family is related to the control of his interpretations. The third family is composed of help functions. These are obviously central in interactive forms of question-answering.

In the first family, control on user actions, we first have the notion of **E-guidance**. This function has quite fuzzy boundaries, it can just include temporal marks guiding the organization of functions, but also manners, durations and instruments which offer a number of details and ways to realize and coordinate actions (*Open the box: use a 2.5 inch key and a screw-driver*). These are deeply related to the argument structure of

the action verb of the instruction (but they are mainly adjuncts or connectors). The next function, **E-framing**, indicates via a statement (often starting a sentence or a paragraph) the context of application of an instruction or of a group of instructions (*[for X25-01 pumps]: disable first...*); it often has the form of a low level title or a frame. Next, **E-expected-result** describes the target result, it is a means for the user to evaluate his performance and to make sure he is on the right track (*at this stage, the sauce must be dark brown*). Finally, **E-elaboration** explains in more depth how to realize an action (*Hold the seedling by the stem with your palm facing the roots of the plant, and turn the soda bottle upside down, [lightly shaking the soil out and the plant with it]*).

The second family is related to the control of the interpretations made by the user. The goal is to make sure he correctly understands the text. In this class fall relatively well-known functions directly associated with rhetorical relations: **E-definition** (gives a definition of a certain concept: *The transmission in your car is a gearbox that transmits powers from the engine to the live axle*), **E-reformulation** (says the same thing with different words or constructions, there is no new informational contents), **E-illustration** (gives one or more examples). Their goal is to ensure that the user correctly understands the terms of the procedure.

The third family is composed of two functions which provide basic help to the user: **E-encouragements** (*at this stage, the most difficult operations have been realized*), **E-evaluation** (*If the paste is really crunchy, then you are an excellent cook you can move on to the next step*).

These functions can be organized by groups leading to forms of explanation plans, e.g. make sure the task is correctly understood (E-definition, E-reformulation) and then warn about risks (E-warning). Planning explanations is the ultimate goal of an author while taking into account the prototypical profile of the readers.

4 Explanation Schemes

Explanation schemes introduce the level of language realizations for explanation functions. These schemes are relatively simple and well-targeted so that they can be used efficiently for producing explanations in question-answering systems.

These schemes are defined from the corpus by coupling EES annotations with the identification of explanation functions in these texts. Generalizing over schemes associated with a given explanation function then allows us to introduce a grammar for each explanation function. So far, the work is entirely done manually. This probably allows a more accurate linguistic analysis than using machine learning techniques. This grammar is based on EES and constrained statements, which form its pre-terminal elements, explanation functions can appear as non-terminal symbols since they may enter into schemes. There is no unique assignment: an explanation scheme may occur in several explanation functions, as can be seen in Walton et al., 2008 for argumentation schemes, with a different perspective.

As an illustration, we give below a few schemes associated with explanation functions (the * indicates multiple occurrences), these are given here in standard bracketed notation,

each bracketed term corresponds to an EES:

E-warning → [warning] / [[warning] [illustration]*] / [[warning] [illustration]* [elaboration]*], etc.

These schemes indicate that an E-warning is composed of the following EES: either a warning alone, or a warning followed by one or more illustrations, or a warning followed by one or more illustrations and elaborations.

E-definition → [[definition] [illustration]*]

E-illustration → [[circumstance] [illustration]*]

E-expected-result → [[circumstance] [statement expr(+modal, +probability)]]

This latter example requires a statement with a modal expression such as 'should' which introduces a probability.

Explanation functions can be complex compounds and may include other explanation functions, e.g.:

E-warning → [[warning] E-illustration* E-elaboration*]

which refers to an EES followed by two explanation functions.

The identification of each EES is based on patterns, induced by corpus analysis. These investigations were realized for each EES separately, on a large corpus of text, besides the 74 texts advocated above. For each EES, the task was to identify and isolate common features, e.g. typical words or classes of words (e.g. negative polarity nouns), morphological aspects, typographic marks, etc. From a global point of view, we developed patterns or rules that account for the structure of each EES (Saito et al., 2006; Miltasaki et al., 2004). These patterns are coded within the framework of <TextCoop> (Saint-Dizier, under submission), which is a platform for discourse analysis based on logic programming with constraints. More patterns and rules are presented in (Bourse and Saint-Dizier, forthcoming 2011).

Since this is ongoing work, it is difficult at this stage to analyse the predictive power of these schemas. Similarly, for the time being they are essentially based on structural considerations, but it is clear to us that communication aspects, some domain as well as general knowledge factor influences the structure of explanations. It is therefore too early to evaluate exactly how these schemas can be used in language generation, although this is our ultimate goal.

5 Conclusion

This paper presents the first stage of a two-level project. We identified various explanation schemes supporting communication goals. The second stage of the project is, given an answer to a question that needs elaborations, how to produce such a response. It is not yet clear to us how this can be done from our schemas, since some form of planning and several types of interactions with knowledge and communication aspects must be considered in the model. Nevertheless, this first step seems to be relatively stable over a number of domains and communication situations. It is suitable for explanation recognition in a number of standard texts. It can also be considered as providing some 'norms' for producing explanations in technical documents.

Most of these structures are now implemented in our <TextCoop> platform (Saint-Dizier, under submission) and

are used for discourse structure processing.

Acknowledgements

This work was partly funded by the Franco-Indian cooperation programme IFCPAR, and by two French ANR research projects: TextCoop and LELIE. We are grateful to the 4 reviewers who made useful suggestions which have been incorporated into this final version.

References

- Bieger, G.R., Glock, M.D. (1984-84). "The Information Content of Picture-text Instructions", *Journal of Experimental Education*, 53, 68-76.
- Carberry, S. (1990). *Plan Recognition in Natural Language Dialogue*. Cambridge: MIT Press.
- Davidson, D. (1980). *Essays on Actions and Events*. Oxford: Oxford University Press.
- Delin, J., Hartley, A., Paris, C., Scott, D., Vander Linden, K. (1994). "Expressing Procedural Relationships in Multilingual Instructions", *Proceedings of the Seventh International Workshop on Natural Language Generation*, 61-70, Maine, USA.
- Delpuch, E., Saint-Dizier, P. (2008). "Investigating the structure of procedural texts for answering How-to Questions", *Language Resources and Evaluation Conference (LREC 2008)*, Marrakech, European Language Resources Association (ELRA).
- Di Eugenio, B., Webber, B.L. (1996). "Pragmatic Overloading in Natural Language Instructions", *International Journal of Expert Systems*, 9 (1), 53-84.
- Fontan, L., Saint-Dizier, P. (2008). "Analyzing the Explanation Structure of Procedural Texts: Dealing with Advices and Warnings", *International Symposium on Text Semantics (STEP 2008)*, Venise, Johan Bos (ed.).
- Keil, F.C., Wilson, R.A. (2002). *Explanation and Cognition*. Cambridge, MA: MIT Press.
- Kosseim, L., Lapalme, G. (2000). "Choosing Rhetorical Structures to Plan Instructional Texts", *Computational Intelligence*, 16 (3), 408-445.
- Lemarie, J., Lorch, R. F., Eyrolle, H., Virbel, J. (2008). "SARA: A Text-Based and Reader-Based Theory of Text Signaling", *Educational Psychologist*, 43, 1-23.
- Longacre, R. (1982). "Discourse Typology in Relation to Language Typology", Sture Allen ed., *Text Processing, Proceeding of Nobel Symposium 51*, Stockholm, Almquist and Wiksell, 457-486.
- Mann, W., Thompson, S. (1988). "Rhetorical Structure Theory: Towards a Functional Theory of Text Organisation", *TEXT*, 8 (3), 243-281.
- Mann, W., Thompson, S. (ed.) (1992). *Discourse Description: Diverse Linguistic Analyses of a Fund Raising Text*. Amsterdam: John Benjamins.
- McKeown, K. (1985). *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*. Cambridge: Cambridge University Press.

Miltasaki, E., Prasad, R., Joshi, A., Webber, B. (2004). "Annotating Discourse Connectives and Their Arguments", NAACL/HLT Workshop on Frontiers in Corpus Annotation, Boston.

Pollock, J.L. (1974). Knowledge and Justification. Princeton: Princeton University Press.

Reed, C. (1998). Generating Arguments in Natural Language. Doctoral thesis. University College London.

Rosner, D., Stede, M. (1992). "Customizing RST for the Automatic Production of Technical Manuals", in R. Dale, E. Hovy, D. Rosner and O. Stock eds., Aspects of Automated Natural Language Generation, Lecture Notes in Artificial Intelligence, 199-214. Berlin: Springer-Verlag.

Saito, M., Yamamoto, K., Sekine, S. (2006). "Using Phrasal Patterns to Identify Discourse Relations", ACL.

Takechi, M., Tokunaga, T., Matsumoto, Y., Tanaka, H. (2003). "Feature Selection in Categorizing Procedural Expressions", The Sixth International Workshop on Information Retrieval with Asian Languages (IRAL2003), 49-56.

Talmy, L. (2001). Towards a Cognitive Semantics, Vol. 1 and 2. Cambridge, MA: MIT Press.

Van der Linden, K. (1993). Speaking of Actions Choosing Rhetorical Status and Grammatical Form in Instructional Text Generation. Doctoral thesis. University of Colorado.

Walton, D., Reed, C., Macagno, F. (2008). Argumentation Schemes. Cambridge: Cambridge University Press.

Webber, B., Di Eugenio, B. (1990). "Free Adjuncts in Natural Language Instruction", Proceedings of Coling 1990, Helsinki, Finland.

Webber, B. (2004). "D-LTAG: Extending Lexicalized TAGs to Discourse", Cognitive Science 28, 751-779.

Wierzbicka, A. (1987). English Speech Act Verbs. Sydney: Academic Press.

Wright, von G.H. (2004). Explanation and Understanding. Ithaca: Cornell University Press.

Zuckerman, I., McConachy, R., Korb, K., (2000), Using Argumentation Strategies in Automated Argument Generation, INLG'2000.

An Exploration of the Linguistic Knowledge for Semantic Relation Extraction in Spanish

Marcos Garcia

Center for Research in Information
Technologies (CITIUS)
University of Santiago de Compostela
marcos.garcia.gonzalez@usc.es

Pablo Gamallo

Center for Research in Information
Technologies (CITIUS)
University of Santiago de Compostela
pablo.gamallo@usc.es

Abstract

A common strategy for Question Answering systems uses high quality ontologies or databases in order to efficiently answer questions. Some approaches to build or enrich these databases rely on machine learning classifiers for obtaining semantically related terms from unstructured text. These classifiers are based on features that may contain several kinds of linguistic knowledge: from orthographic or lexical information to more complex features, including PoS-tags, syntactic dependencies or semantic information. In this paper we select four main types of linguistic features and systematically evaluate their performance in semantic Relation Extraction. Although the combination of some types of linguistic features allows to improve the f-score of the classifiers, we observed that adjusting the positive/negative ratio of the training examples, we can build high quality classifiers with just a single type of linguistic feature, based on generic lexico-syntactic patterns. Experiments were carried out with the Spanish version of Wikipedia.

1 Introduction

Recent approaches for Question Answering systems use high quality ontologies or databases for answering questions. Building these resources manually is time-consuming, so the improvement of Information Extraction (IE) methods becomes a crucial task. In this sense, Relation Extraction (RE) attempts to automatically obtain semantic information related to named entities from unstructured text, which, in turn, is incorporated into machine readable databases and ontologies. For instance, from the sentence *Amancio Ortega (born March 28, 1936) is a Spanish fashion entrepreneur*, a system may learn different properties about Amancio Ortega: his birth-date, his nationality as well as his occupation.

Most of recent RE techniques employ machine learning algorithms for extracting information. They arrange into a set of linguistic features the contexts or sentences in which pairs of related entities occur. These features are then used to train a classifier. The starting point is a number of labeled training examples (e.g., relations “Amancio Ortega - fashion entrepreneur”, “Billie Holiday - singer”...), as well as a corpus-

based strategy to search, identify and represent as features those sentences (contexts) in which the training examples occur. These strategies may have different degrees of supervision, from manually constructed corpora to unsupervised methods that do not need human intervention for revising the sentences. Moreover, it must be pointed out that linguistic features can be represented at different levels of generality, according to different types of linguistic information. However, there are no much work on the importance of knowing what linguistic knowledge is actually useful in order to increase the performance of these systems.

In this article, our aim is to evaluate the impact of different types of linguistic features in the RE task for Spanish. For this purpose, a distant supervised learning system was built and tested with different types of features: bags of lemmas and PoS-tags, lexico-syntactic patterns and syntactic dependencies. We evaluated three main aspects concerning extraction of semantic relations: First, we measured the impact of linguistic information by training several classifiers that only differ in the type of linguistic knowledge used to define their features. Second, we analyzed the best feature combinations. And third, we found the best ratio of positive and negative examples according to each type of linguistic features.

Preliminary results show that, usually, the best performance was achieved using just a single type of features based on generic lexico-syntactic patterns, built by means of a *longest common string* algorithm. However, the combination of some types of linguistic features allows to increase recall without losing precision. Furthermore, we also observed that a deep analysis of the positive/negative ratio of the training examples improves the f-score of the classifiers.

In Section 2 we show the related work. Section 3 explains the method for obtaining the data used to build the classifiers. In Section 4 we present the feature space as well as the different types of features used. Finally, Sections 5 and 6 show the results and the future directions of our work.

2 Related Work

Since the work of Hearst [1992], many approaches have been implemented in order to obtain patterns for extracting related terms, such as Brin [1998] or Ravichandran and Hovy [2002].

In the recent years, works like Mann [2002], Fleischman *et al.* [2003] or Agichtein [2005] use different features for training machine learning classifiers with several algorithms.

The selected features differ according to the linguistic information they process (word, lemma, PoS-tag, orthographic form, etc.). Other works [Bunescu and Mooney, 2005; Suchanek *et al.*, 2006; Nguyen *et al.*, 2007] created statistical models using the results of syntactic parsing. Despite that some preliminary results of these researches show that the use of deep linguistic knowledge is better for relation extraction, Bunescu and Mooney [2005] warn of the importance of knowing what of this linguistic information is actually useful in order to increase the performance of an IE system.

In this sense, some works must be cited: Kambhatla [2004], Zhou *et al.* [2005] and Jiang and Zhai [2007], which evaluate the effectiveness of diverse lexical, syntactic, and semantic information in RE, and Zhao and Grishman [2005] which use a more complex kernel-based strategy to combine features of different linguistic levels. However, their work differs from ours in a key point: the linguistic feature space is not the same, in particular they do not make use of lexico-syntactic patterns. Moreover, our evaluation concerns other languages than English as well as a deep analysis of the impact of negative examples on the training data.

Finally, Wu and Weld [2010] present *woe*, an Open Information Extraction method based on data obtained from the semi-structured resources of Wikipedia.

3 Method Overview

In order to easily evaluate the performance of various types of features, we first obtain an annotated corpora, used to build the different classifiers. The method is the following:

A Wikipedia snapshot is converted into plain text, removing external links as well as formatting marks. Given a semantic relation, for instance *Occupation*, we get a large set of related pairs from Wikipedia infoboxes: e.g., “John Lennon - musician”, “John Lennon - composer”, “Fernando Pessoa - poet” (with a precision of about 95%). We use these pairs to select from the plain text of Wikipedia sentences that contain both a named entity and an occupation, so no bootstrapping is required. If the two terms match a known pair of the initial list, the example is annotated as positive. Otherwise, it is annotated as negative.

Then, we use FreeLing [Padró *et al.*, 2010] to lemmatize, PoS-tag and recognize the proper names in the sentences. Syntactic dependencies are identified with a robust, partial, and rule-based parser [Gamallo and González, 2011].

The two target entities are replaced by both **X** and **Y** (standing for the first and the second entities of the pair, respectively) and put labels to mark the left, middle, and right contexts. The middle context has a maximum size of 12 tokens, while the other ones will not exceed 3 tokens. These window size values were empirically set after having tested different values in preliminary experiments.

All the process is performed without human revision, so it may lead us to automatically annotate *false positives* (“*Linus Torvalds* discussed with a *software engineer* in Italy”, **true**) or *false negatives* (“*Fernando Pessoa* was a *literary critic*”, **false**, since this attribute does not appear in the infobox). The manual revision of the test set showed that this method has a precision of about 80%. This issue will be addressed by using

Sentence: Amancio Ortega Gaona is a Spanish fashion entrepreneur.

Structure:

pos	token	lemma	tag	head	label
0	X	X	NP	1	subj
1	is	be	V	0	-
2	a	a	DI	5	spec
3	Spanish	spanish	ADJ	5	modif
4	fashion	fashion	N	5	modif
5	Y	Y	N	0	attr

Figure 1: Example of a sentence with its Linguistic Structure.

machine learning algorithms (e.g. SVM) that are tolerant to noise by minimizing the effect of the false samples [Aslam and Decatur, 1996].

4 Feature Space and Types of Features

Each selected, tagged, and parsed sentence represents a *linguistic structure* containing all the relevant information required by the relation extraction systems. A linguistic structure or parsed sentence can be conceived as a knowledge-rich space incorporating several levels of linguistic information. This space should be as complete as possible in the sense that all features potentially useful for RE are included.

A linguistic structure contains the surrounding context of the related entities: **X** stands for the named entity and **Y** for the occupation name. We include within a linguistic structure the left context of the first entity, the middle context, and the right context of the second entity. Left and right contexts have a maximum size of 3 tokens (from *pos* -1 to -3 and +1 to +3). Middle context may contain 12 tokens (from 1 to 12). The window size was empirically set to 3 and 12 after having tested different values in preliminary experiments. We will distinguish experiments using all contexts (left, right, and middle) from those considering only the middle one.

In Figure 1, columns 1, 2, 3, and 4 respectively stands for position, token, lemma, and PoS-tag. This information was obtained with the lemmatizer and PoS-tagger included in FreeLing. Furthermore, the structure also contains the syntactic dependencies identified by the parser. Column 5 identifies the head position of the current token, while the label of the dependency is shown by column 6. Since we use a partial parser, not all possible word dependencies are identified. Our structure was inspired by the output parsing format defined in [Lin, 1998], adopted by learning tasks of CoNLL.

These linguistic structures are used to extract the different types of linguistic features needed by our classifiers. We use four main different types of linguistic features in order to build the relation extraction systems:

Basic Patterns: The first type of feature uses all the explicit information contained in the linguistic structures except two elements: dependency relations and some lemmas. We only take into account lemmas of verbs, common nouns and prepositions. We have observed in preliminary experiments that the performance of classifiers decreased when either these type

of lemmas were removed or all lemmas including grammatical words (stop words), adjectives and proper names were retained. It follows that verbs, common nouns, prepositions are critical pieces of information to define the lexico-syntactic contexts of the target terms. An example of basic patterns associated to the relation “Occupation” is the following:

Sentence: Amancio Ortega Gaona is a Spanish fashion entrepreneur.

Polarity: Amancio Ortega Gaona Occupation entrepreneur, **true**.

Pattern: <X be_V DI ADJ fashion_N Y>

(where V means verb, DI indefinite article, ADJ adjective, and N common noun). Let us note that basic lexico-syntactic patterns require a huge training corpus, due to their lack of flexibility. Small variations in punctuation, adjective or adverb modification, determiners, etc. will generate different features. In this case, the problem of sparse data is crucial.

Pattern Generalization: To minimize the sparse data problem, we applied an algorithm based on similarity between basic patterns, generalizing them and therefore increasing the coverage of the model. In order to generalize two patterns, we check first if they are similar and then all those units that they do not share are removed [M. Ruiz-Casado and E. Alfonso and P. Castells, 2005]. The similarity, noted *Dice_Lcs*, between two patterns p_1 and p_2 is defined using the longest common string and Dice metric as follows:

$$Dice_Lcs(p_1, p_2) = \frac{2 * lcs(p_1, p_2)}{length(p_1) + length(p_2)} \quad (1)$$

where $lcs(p_1, p_2)$ is the size of the longest common string between patterns p_1 and p_2 , and $length(p_i)$ represents the size of pattern p_i . It means the similarity between two patterns is a function of their longest common string and their lengths.

After computing the similarity between two patterns p_1 and p_2 , the longest common string is extracted if and only if p_2 is the most similar pattern of p_1 and the similarity score is higher than a particular threshold (0.75 in our tests). The longest common string of two patterns is considered as the generalized pattern out of them.

Bags of Lemmas and Tags: Instead of using a set of entire patterns as features, a common method that allows to increase the coverage of the system is the use of smaller items, such as lemmas with PoS-tags. In this case, the sentence *Amancio Ortega Gaona is a Spanish fashion entrepreneur* would generate the following features (notice again that for some categories only the tag is retained): <be_V>, <DI>, <ADJ> and <fashion_N>. The use of negative examples is probably more important here, allowing the algorithm to learn which lemmas are crucial for making a correct decision. These classifiers are then close to the baseline bag-of-words models.

Syntactic Dependencies: Dependency information is obtained by the referred robust, multilingual, and rule-based

parser. The analysis of the parser is partial, but it provides the most frequent dependencies between the target terms and the units within the patterns. Here again only the lemmas of verbs, common nouns, and prepositions are retained.

We consider a subset of all the syntactic dependencies derived from the full linguistic structures. Given a sentence, two types of dependencies are retained:

- dependencies between the two target terms (X or Y),
- dependencies between one of the two target terms and one entity of the (left, middle, or right) context.

For instance, from the sentence *Amancio Ortega Gaona is a Spanish fashion entrepreneur*, the selected dependencies would be the following: <subj;X;be_V>, <attr;be_V;Y>, <spec;Y;DI>, <modif;Y;ADJ> and <modif;Y;fashion_N>.

Each feature is a triplet constituted by the dependency label, the head, and the dependent expression. Only dependencies with at least one term (X or Y) was selected from the linguistic structure. The selected information, thus, corresponds to the local dependency context around the related terms.

Note that the analysis is very partial. In many cases, the parser is not able to complete the dependency path between the two terms. Rule-based grammars for other languages than English are often not complete or they are not freely available. Finally, labels of dependencies (e.g., modifier, specifier, subject, etc.) are also taken into account to define the features.

5 Experiments

We evaluated both the performance of the features individually as well as the best combinations of them. We also examined the effect of limited training input on the learning process by incrementally adding examples to the training data. Finally, we analyzed the impact of negative examples in the training set.

The experiments were performed with WEKA [Witten and Frank, 2002], using its implementation of the SMO algorithm [Platt, 1999]. We made this choice because in preliminary experiments (using Naive Bayes, Decision Tree as well as SMO algorithms), SMO scored the best.

5.1 Initial Data

The training examples were obtained from the Spanish Wikipedia (May 2010) with the method showed in Section 3. We focused on examples of the relations “Occupation” (a kind of *is_a* relation) and “Birth Place”. We first selected about 50,000 relation pairs for each relation from infoboxes. Then, we identified near 600,000 sentences containing a named entity and an occupation or a location, which were automatically classified as positive or negative. Finally, we randomly selected an initial set of 2000 sentences (for each relation) for training, 50% of them being positive examples.

For testing, we manually revised two randomly selected sets of about 700 and 500 different sentences for “Occupation” and “Birth Place” relations, respectively.

Model	Prec.	Rec.	f-score
<i>pattern-all</i>	100%	12.99%	23%
<i>pattern-mid</i>	98.04%	56.5%	71.68%
<i>pattern_gen-mid</i>	97.72%	72.6%	83.31%
<i>bow-all</i>	83.02%	62.15%	71.08%
<i>bow-mid</i>	88.28%	59.6%	71.16%
<i>dep-all</i>	86.62%	65.82%	74.8%
<i>dep-mid</i>	86.21%	35.31%	50.1%

Table 1: Precision, Recall and f-score of 7 classifiers trained with 2000 sentences (“Occupation” relation).

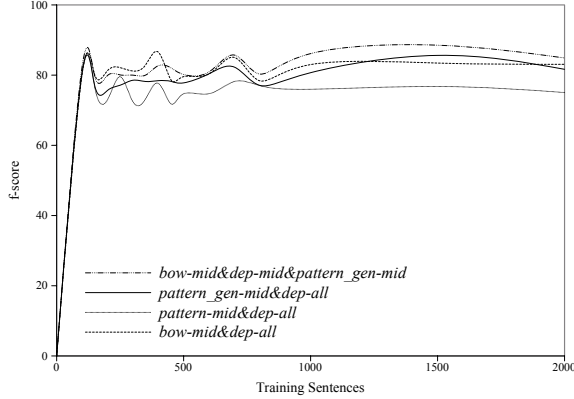


Figure 2: f-score vs training size of the 4 best combinations of features (“Occupation” relation).

5.2 Results

Table 1 shows the results obtained with 7 types of linguistic features, extracted from training set of 2000 linguistic structures. The types of features are those explained in Section 4: *pattern-all* and *pattern-mid* use the basic patterns extracted from the linguistic structures as features. The former was trained with all contexts (left, right, and middle), while the latter was only trained with the middle context.

pattern_gen-mid uses as features the generalized patterns and the middle context.

bow-all and *bow-mid* were built with the bag of lemmas and tags technique.

dep-all and *dep-mid* are the dependency-based models described in the previous section.

Precision is the number of correct positive decisions divided by the number of positive decisions. Recall here refers to the number of correct positive decisions divided by the total number of positive examples in the test set.

Table 1 let us observe that the best features are those based on lexico-syntactic patterns with *mid* contexts: *pattern_gen-mid* and *pattern-mid*, with f-score values between 71%-83%. The score reached by *pattern-all* is much lower because of very poor recall values. This is due to the fact that, in this case, both left and right contexts tend to be too sparse. Table 1 also shows that features based on dependencies and on bags of lemmas and tags achieve similar precision (83%-86%), and that in these cases, *all* contexts scored better than *mid*.

The results of these tests allow us to know the performance of the individual types of features. The next experiment at-

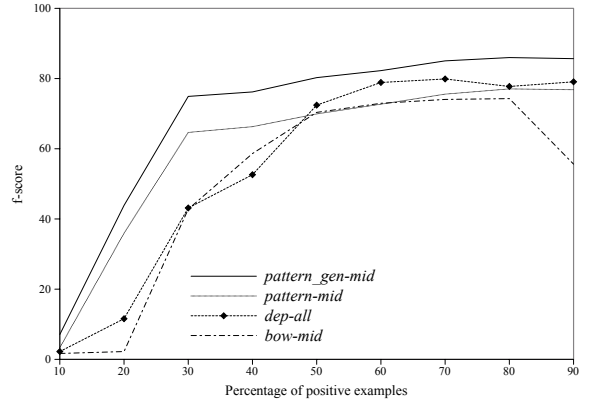


Figure 3: f-score vs ratio of positive-negative examples of the best classifiers for each linguistic type trained with 500 sentences (“Occupation” relation).

tempts to know the performance of several combinations of individual features, as well as their learning curve.

When analyzing the differences between the models, we both compute the Dice similarity coefficient and a simple count statistic aimed to find whether the decisions taken by two models are or not on the same instances. In general, a high Dice coefficient may imply that there are few correct decisions taken on different instances and, conversely, a low Dice coefficient means that there are many correct decisions taken on different instances. Only not very similar models were combined since they are likely to be complementary.

The results of several combinations based on a similarity analysis show that these classifiers may help to achieve a trade-off between precision and recall. Figure 2 shows the f-score curve of the four best combinations. Notice that the best combinations scored better than the best individual feature (*pattern_gen-mid*, which reached 83.31%), achieving values of 88% with around 1500 of training sentences. However, it is important to note that the best f-score values are reached without making use of the whole training set.

The initial set of 2000 sentences, with which we performed the previous experiments, had a 50%/50% ratio of positive and negative examples. Our next experiment concerns the evaluation of the positive/negative (P/N) ratio of the training examples for each type of linguistic feature.

In order to know the best P/N distribution, we collected several sets of sentences differing in the percentage of positive and negative examples they have. Note that this evaluation also deals with the amount of P/N instances, and not only with its P/N ratio. So, in order to avoid this effect, we performed two major experiments: (i) we collected nine sets of 500 sentences differing in the P/N ratio: from 10%/90% to 90%/10% and (ii) we did the same distributional partitions from a larger corpus (sets of 2000 sentences). Finally, we analyzed how the learning process is influenced by the P/N ratio as well as by the number of positive and negative examples.

Figures 3 and 4 show the f-score curve of the best classifiers for each main type of features, depending the P/N ratio of the training set. Classifiers from Figure 3 were trained with nine partitions of 500 sentences, while in Figure 4 the

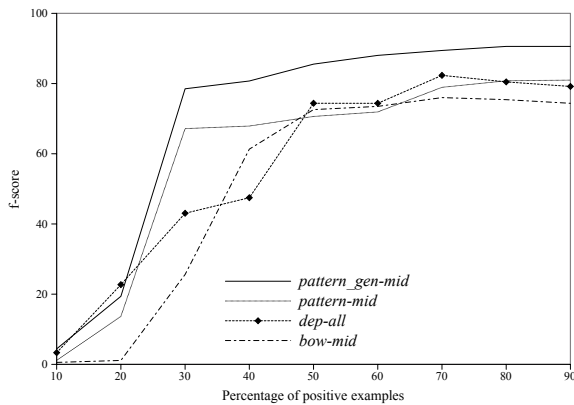


Figure 4: f-score vs ratio positive-negative examples of the best classifiers for each linguistic type trained with 2000 sentences (“Occupation” relation).

Model	Prec.	Rec.	f-score	Diff.
<i>pattern-all</i>	100%	14.56%	25.42%	+2.4
<i>pattern-mid</i>	95.23%	65.93%	77.92%	+6.2
<i>pat_gen-mid</i>	94.95%	82.69%	88.39%	+5.1
<i>bow-all</i>	72.86%	95.87%	82.79%	+11.7
<i>bow-mid</i>	82.33%	64.01%	72.02%	+0.9
<i>dep-all</i>	77.89%	81.31%	79.56%	+4.8
<i>dep-mid</i>	73.98%	60.16%	66.36%	+16.3

Table 2: Precision, Recall and f-score of 7 classifiers trained with 2000 sentences (“Occupation” relation). The positive/negative ratio was adjusted for each model. *Diff.* shows the f-score difference with 50%/50% models.

training sets had 2000 sentences. In most cases, the peak of the f-score curve is reached with between 60% and 80% of positive data (except for patterns-based features, whose performance gradually improves with more positive examples). This is true for the two tests (500 and 2000 sentences), so we can infer the best P/N ratio for each type of feature. Note that the 20%/80% ratio in Figure 4 has the same number of positive examples than the 90%/10% in Figure 3.

Pattern-based features need a large set of training data in order to learn a enough set of patterns as well as their confidence. However, Figures 3 and 4 show that these models also need a low ratio of negative examples (its performance becomes stable with about 80%/20%). By contrast, *dep-all* (and also *dep-mid*) achieves its best score with 70%/30%, as it decreases with less negative data. Finally, features based on bags of lemmas are more variable, and work better with, at least, 20% of negative examples (but never with less than 50% of positive training data).

The results from the previous experiment provide us information to train new classifiers with the P/N ratio adjusted to each type of feature. So, we built seven new classifiers based on the same seven individual types of features described above. We randomly selected seven sets of 2000 sentences, each one with a distribution of positive and negative examples adjusted to the needs of each type of features, and tested them on the 700 manually revised examples.

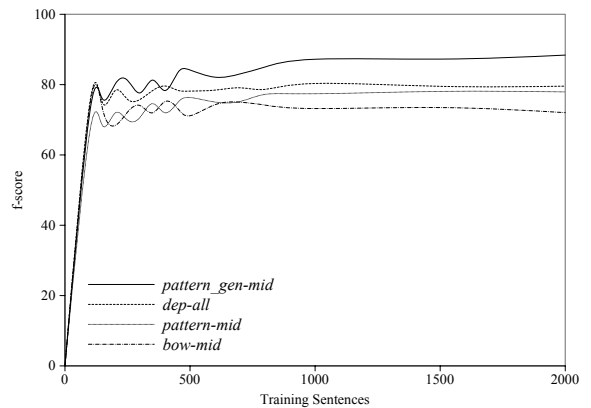


Figure 5: f-score vs training sentences of the best classifiers for each linguistic feature (“Occupation” relation). Positive/negative ratio was adjusted for each classifier.

Model	Prec.	Rec.	f-score
<i>pattern-all</i>	88.88%	13.55%	23.52%
<i>pattern-mid</i>	76.47%	44.06%	55.91%
<i>pat_gen-mid</i>	88%	74.57%	80.73%
<i>bow-all</i>	42.6%	83.05%	56.32%
<i>dep-all</i>	49.36%	66.1%	56.52%

Table 3: Precision, Recall and f-score of 5 classifiers trained with 2000 sentences (“Birth Place” relation). The positive/negative ratio was adjusted for each classifier.

Figure 5, shows the f-score of the best individual features (for each one of the 4 main types) in different partitions. The P/N ratio in the training set was adjusted for each classifier. Note that the performance of all classifiers was improved (namely due to the improvement in recall) from those obtained with the 50%/50% P/N training corpus (4th column in Tables 1 and 2). Furthermore, it can be observed that the curve stabilizes when the training corpus is constituted by about 1000 sentences. So, no more training corpus is required to improve results.

Finally, Table 3 contains the results obtained from an ongoing test with “Birth Place” semantic relation. Unlike what occurred with “Occupation”, *bow* models (and also *dep*) achieved lower precision values. However, results still follow the general tendencies. Note that the generalization of lexico-syntactic patterns also achieves high f-score values (80.73%) in these experiments. With both semantic relations, the performance of the best classifiers based on individual types of features, built on the analysis of the best P/N ratios, is similar than the best combinations.

6 Conclusions and Further Work

The experiments described in this article can help us to outline some ideas regarding the impact of the linguistic knowledge on Relation Extraction for Spanish.

First, the adaptation of the positive/negative ratio in the training set helps to improve the performance of the classifiers. We observed that features based on patterns of PoS-

tags and (some) lemmas need a high percentage of positive examples. Those based on bags of lemmas and syntactic dependencies perform better with more than 50% of positive examples, but they need at least 20% of negative data.

Second, the application of the *longest common string* over the lemma_tag patterns are good features to define models with high precision and relatively good recall. Models built of bags of lemmas and tags as well as dependency-based show more variation in precision values. Concerning these dependency models, we have to note that the use of *partial parsing* may also involve variations in recall.

If the objective is to obtain a trade-off between precision and recall, we can analyze the similarity between the results of several types of features in order to choose the best combinations for a specific relation. However, if we attempt to rapidly build new RE systems, the construction and generalization of patterns with PoS-tags and lemmas is the best solution.

Further experiments will be focused on the construction of new classifiers based on combinations of the best types of features, each one with its positive/negative ratio adjusted. Moreover, we are testing the impact of noisy data in the training phase, as well as the adaptation of our strategy for extracting other semantic relations and text typologies.

Acknowledgments

This work has been supported by the MICINN, within the project with reference FFI2010-14986.

References

- [Agichtein, 2005] E. Agichtein. *Extracting Relations from Large Text Collections*. PhD thesis, Columbia University, 2005.
- [Aslam and Decatur, 1996] J. A. Aslam and S. E. Decatur. On the sample complexity of noise-tolerant learning. *Information Processing Letters*, 57:189–195, 1996.
- [Brin, 1998] S. Brin. Extracting patterns and relations from the world wide web. In *WebDB Workshop at 6th International Conference on Extending Database Technology*, pages 172–183, 1998.
- [Bunescu and Mooney, 2005] R. C. Bunescu and R. J. Mooney. A Shortest Path Dependency Kernel for Relation Extraction. In *Proceedings of the HLTC on EMNLP*, pages 724–731, 2005.
- [Fleischman et al., 2003] M.I. Fleischman, E. Hovy, and A. Echihiabi. Offline strategies for online question answering: Answering questions before they are asked. In *Proceedings of the 41st Annual Meeting of the ACL*, pages 1–7, 2003.
- [Gamallo and González, 2011] P. Gamallo and I. González. A Grammatical Formalism based on Patterns of Part-of-Speech Tags. *Journal of Corpus Linguistics*, 16(1):45–71, 2011.
- [Hearst, 1992] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics*, volume 2, pages 539–545, Morristown, NJ, USA, 1992.
- [Jiang and Zhai, 2007] J. Jiang and C. Zhai. A Systematic Exploration of the Feature Space for Relation Extraction. In *Proceedings of NAACL/HLT*, pages 113–120, 2007.
- [Kambhatla, 2004] N. Kambhatla. Combining lexical, syntactic and semantic features with Maximum Entropy models for extracting relations. In *Proceedings of the 42th Annual Meeting of the ACL*, 2004.
- [Lin, 1998] Dekang Lin. Dependency-based Evaluation of MINIPAR. In *Proceedings of the Workshop on the Evaluation of Parsing Systems*, Granada, 1998.
- [M. Ruiz-Casado and E. Alfonseca and P. Castells, 2005] M. Ruiz-Casado and E. Alfonseca and P. Castells. Automatic assignment of Wikipedia encyclopedic entries to WordNet synsets. *Proceedings of the Atlantic Web Intelligence Conference*, 3528:380–386, 2005.
- [Mann, 2002] G. S. Mann. Fine-Grained Proper Noun Ontologies for Question Answering. In *SemaNet’02: Building and Using Semantic Networks*, Taipei, Taiwan, 2002.
- [Nguyen et al., 2007] Dat P. T. Nguyen, Y. Matsuo, and M. Ishizuka. Relation Extraction from Wikipedia Using Subtree Mining. In *Proceedings of the 22nd national conference on Artificial intelligence*, volume 2, pages 1414–1420, 2007.
- [Padró et al., 2010] Ll. Padró, M. Collado, S. Reese, M. Lloberes, and I. Castellón. FreeLing 2.1: Five Years of Open-Source Language Processing Tools. In *Proceedings of the 7th Language Resources and Evaluation Conference*, La Valletta, Malta, 2010.
- [Platt, 1999] J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in kernel methods: support vector learning*, pages 185–208, Cambridge, MA, USA, 1999.
- [Ravichandran and Hovy, 2002] D. Ravichandran and E. Hovy. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting of the ACL*, pages 41–47, 2002.
- [Suchanek et al., 2006] F. M. Suchanek, G. Ifrim, and G. Weikum. LEILA: Learning to Extract Information by Linguistic Analysis. In *Second Workshop on Ontology Population at ACL/COLING*, 2006.
- [Witten and Frank, 2002] I. Witten and E. Frank. Data mining: practical machine learning tools and techniques with Java implementations. *SIGMOD Rec.*, 31:76–77, 2002.
- [Wu and Weld, 2010] F. Wu and D. S. Weld. Open information extraction using Wikipedia. In *Proceedings of the 48th Annual Meeting of the ACL*, pages 118–127, 2010.
- [Zhao and Grishman, 2005] S. Zhao and R. Grishman. Extracting Relations with Integrated Information Using Kernel Methods. In *Proceedings of the 43rd Annual Meeting of the ACL*, pages 419–426, 2005.
- [Zhou et al., 2005] G. Zhou, J. Su, J. Zhang, and M. Zhang. Exploring Various Knowledge in Relation Extraction. In *Proceedings of the 43rd Annual Meeting of the ACL*, pages 427–434, 2005.

Learning Quasi-Similar Pairs for Social Roles

Octavian Popescu, Emanuele Pianta, Bernardo Magnini
{popescu, pianta, magnini}@fbk.eu

Abstract

The experience from the Textual Entailment contests has revealed that an important part of entailment relationships requires extensive world knowledge, which is unlikely to be found in a repository, because this information is bound to specific contexts. In this paper we present an unsupervised and language independent method of acquiring quasi-similar pairs which capture this particular type of knowledge.

1 Introduction¹

Textual Entailment (TE) is the task of analyzing textual pairs, $\langle T, H \rangle$, and to decide whether T entails H [Dagan, Glickman, & Magnini, 2006]. The entailment relationship between two pieces of text, T and H , is understood as the fact that most people reading H , after firstly reading T , would conclude that the information in H has already been transmitted in T . The TE task is considered an important Natural Language task with a broad application scope: question answering [Lin&Pantel 2001], information summarization [Hovy2003], document collection [Voorhes 2001], sentence compression [Cohn&Lapata 2009], language generation [Deleger & Zweigenbaum, 2009].

In its most general form, there is no restriction on the form or on the semantic content of the $\langle T, H \rangle$ pair. In some cases, deep semantic inferences are required by the entailment relationship. In fact, the experience accumulated in the TE contests [Giampiccolo 2008] has shown that the majority of the inferences is decidable only by using a significant amount of knowledge. In particular, the so-called “human understanding of lexical and world knowledge” [Clark et al. 2007] plays a crucial role in the inference decision. In example [1]

(1T) L’ala Benzoni ha firmato col Milan.
the wing Benzoni signed for Milan.

(1H) Il giocatore Benzoni gioca per i rossoneri.
the player Benzoni plays for rossoneri.

$1T$ entails $1H$. But the entailment relationship in the pair in (1) is not immediately obvious. An automatic system must somehow know that in Italian *ala* may refer to a particular type of player and that *rossoneri* is the common nickname of the Milan football team. The essential information is that the elements of the pairs [*ala*, *giocatore*] and [*Milan*, *rossoneri*] are quasi-similar in the Italian football context. We call such pairs quasi-similar because their similarity is bound to a very specific domain.

Example (1) is non limitative; similar entailment relationships can be inferred only if specific information about how the world is seen in particular environments is available. In particular, an important part of this information is made up of knowledge regarding social roles, where a social role word is loosely defined as a word that is used to refer to a person, or a group of persons, by referring to an occupation, hobby, activity etc. These social roles are bound to a social context, and they are not universal. Nevertheless they are very frequent in normal communication and their proper understanding is essential for TE.

The problem with this type of entailment is threefold. Firstly, the required information is not found in computational dictionaries like WorldNet, LDOCE, Roget’s thesaurus etc. Secondly, this information is language specific, and it is not part of a general ontology. Thirdly, the quasi-similar pairs are specific to a certain social context and period and so, they need to be learned continuously and independently from language to language.

One way to solve this problem is to learn the quasi-similar pairs from a large corpus.

In this paper we present an unsupervised, language independent method of learning the world knowledge regarding the social roles needed in textual entailment. The method scans a large corpus, which has been previously annotated by a Named Entity Recognizer. Our system takes advantage of the fact that the same entity is mentioned using different social roles. By relying on the results of coreferencing certain entities with a very high accuracy, the social roles that are quasi similar are extracted. We accomplish this by using a shallow context analysis and a statistical method.

¹ This work is partially supported by the BCROCE project

We used the Adige500k corpus, a corpus made of pieces of news from a local Italian newspaper [Magnini et. al 2006]. Our method works best on news corpora because it brings three main advantages for our goal of acquiring social roles: 1) the news industry is concerned with social roles 2) the pieces of news represent compact, independent information and 3) the journalistic style repels the repetition and it prefers that the same entities be mentioned using different words. Nevertheless, our method would work on any corpus.

In Section 2 we review the relevant literature. In Section 3 we present the details of learning the quasi-similar words denoting social professions. In Section 4 we present the results of evaluating the accuracy of the system. The Conclusion and Further Research section concludes this paper.

2 Related Work

Some unsupervised methods of acquiring lexical knowledge for textual entailment have been developed in order to take advantage of the similarity of the syntactic context [Lin&Pantel 2001; Dagan et al. 2008]. These methods process the output of a parser. The dependency of a parser brings forward three main disadvantages: 1) parsing is a time consuming process, 2) parsers are not language independent tools and many languages lack one, 3) the accuracy of the parser is an upper bound for the accuracy of the whole system.

The bootstrapping methods have been used in textual entailment in specialized corpora, like medical ones [Jacquemin&Bourigault, 2003]. However, the relationships between words denoting social roles which are quasi synonyms, are very heterogeneous and there is no clear way in which the bootstrapping might be applied in these particular learning tasks.

The vectorial model has also been employed for textual entailment [Hovy et. al 2007; Szpektor et. al 2008]. The contexts are modeled as vectors and rules of entailment can be applied only in similar contexts, where similarity is computed according to a similarity metrics. The similarity metrics can work on the dependency trees as well [Kouleykov et. al. 2005].

One of the problems for the TE systems is the difficulty of creating contradiction samples. A contradictory pair is formed by two sentences that cannot possibly be true at the same time. To create a contradictory corpus is difficult, as the contradiction relationship seems to require a deeper analysis than the entailment relationship. Our method allows a natural extension which can be used to create both entailment and contradiction examples involving social roles.

The method of acquiring the quasi-similar pairs we present is instrumental in building a contradiction corpus.

A method for finding synonyms has been applied in order to simplify the technical texts [Elhadad&Sutaria 2007]. The method does not allow the learning from corpus and it can paraphrase only the technical expressions containing words whose synonyms set is already known.

Our method for the acquisition of quasi-similar pairs uses only shallow text processing. The context of interest is determined in constant time around the entity of interest.

3 Social Roles and News Local Coreference

The Adige 500k consists of a collection of individual pieces of news that extend approximately over seven years. In Table 1 we present a fragment from a piece of news.

Italian	English
... Missione cinese per l' arcivescovo di Trento, Luigi Bressan. Il soggiorno in Cina di <i>monsignor</i> Bressan si inserisce in un ampio progetto di	... Chinese Mission for the <i>arcivescovo</i> of Trento, Luigi Bressan. The travel of <i>monsignor</i> Bressan is part of a big project

Table 1. Fragment of a Piece of News

In this fragment of an ordinary piece of news there are two name mentions, *Luigi Bressan* and *Bressan* respectively. We present below a coreference technique of name mentions that is extremely accurate, over 99.7% accuracy. A side effect of coreferencing the name mentions is that we have direct access to the noun phrases that include the respective name mentions. Some of these noun phrases carry the quasi-similar pairs of social roles. In Table 1 we see that *arcivescovo* and *monsignor* are quasi-similar pairs.

The method of finding the quasi-similar pairs is based on coreferencing the name mentions, finding the extended noun phrases and extracting potentially quasi-similar words, and filtering the potential candidates for obtaining the final quasi-similar pairs.

The coreference and the filter modules implement a statistic approach. No additional data, except from the one already existing in the corpus itself is needed. By employing linear rules of stopping words, we recognize the noun phrases. Optionally, we can start with a set of words denoting social roles using SUMO [Niles 2003]. Taking advantage of the fact that SUMO is aligned to Wordnet, which is available in some European languages, like English, Italian, Portuguese, Rumanian, Spanish etc., [Pianta et. al. 2002] a part of the social roles can be available to start with. The processes of finding the quasi-similar pairs for these words and for all the others that are to be found in the corpus, is the same as above.

In Section 3.1 we present the set of social role words we are interested in. Based on I-CAB [Pianta et. al. 2006], we present the distribution of these words in corpus. We also present here the method of pairing the social roles once the coreference has been established. In Section 3.3 we look at the pairs of quasi-similar social roles extracted.

3.1 I-CAB

From the Adige500k corpus a set consisting of all the news in two pairs of two consecutive days each has been manually annotated, creating the I-CAB corpus [Magnini et. al. 2006].

After the annotation of mentions of type PERSON, each mention was additionally annotated in order to individuate the semantic information expressed by the mention regarding a specific entity. The process of deciding which phrases are relevant and at the same time, instrumental in the computational entailments, was iterative [Pianta et. al. 2006]. The following three attributes, Activity, Role, and Affiliation are not independent of each other. Activity refers to the actual activity performed by the person, while Role refers to the position they occupy. Each group of these three attributes is associated with a mention and all the information within a group has to be derived from the same mention. If different pieces of information derive from distinct mentions, we will have two separate groups. For instance, the three co-referring mentions “the journalist of Radio Liberty”, “the redactor of breaking news”, and “a spare time astronomer” lead to three different groups of Activity, Role and Affiliation.

attribute	Values
FIRST_NAME	<i>Ralph, Greg</i>
MIDDLE_NAME	<i>J., W.</i>
LAST_NAME	<i>McCarthy, Newton</i>
NICKNAME	<i>Spider, Enigmista</i>
TITLE	<i>Prof., Mr.</i>
GENDER	<i>F, M</i>
ACTIVITY	<i>author, doctor</i>
AFFILIATION	<i>The New York Times</i>
ROLE	<i>manager, president</i>
PROVENANCE	<i>South American</i>
FAMILY_RELATION	<i>father, cousin</i>
AGE_CATEGORY	<i>boy, girl</i>
HONORARY	<i>the world champion 2000</i>

Table 2. Ontological Attribute Description

The social roles are very well represented in the I-CAB, and because this corpus represents a random, sufficiently large sample of the Adige500k, we can assume that the percentages are similar in the whole corpus. In Table 3 we present these figures.

attribute	#mentions	different
ACTIVITY	973 (13%)	322
AFFILIATION	566 (7%)	389

attribute	#mentions	different
ROLE	531 (7%)	211
HONORARY	69 (2%)	63
MISCELLANEA	278 (3%)	270

Table 3. Distribution of the Attributes in Corpus

3.2 Local Coreference

The Adige 500k is divided into pieces of news and within each of them the mentions containing personal names has been individuated. The task of clustering the personal name mentions according to the entity they refer to is a complex one. However, it has been shown that restricting the coreference at the news level, the one entity per document hypothesis is valid [Popescu&Magnini 2007]. With an accuracy of over 99% the same name refers to only one person within a piece of news. The names are considered with their variants, because a person could be mentioned using just the first or last name, or using just abbreviations.

In order to make all the possible coreferences within a piece of news we implemented a system of finding the best match for a name variant. For each name a list of possible candidates is computed; for example for a first name a list of possible last names is computed. Let us suppose that the candidate list has cardinality bigger than two and that we search for a completion for a first name. The list of candidates contains the last_name - labeled name mentions. We consider for each first_name (l) and last name (L), the probability of joint occurrence, and the probability of common occurrence in a name (l_L). These numbers have been calculated with respect to each topic (t) independently. The probabilistic equation which involves these variables is:

$$\begin{aligned}
 the_most_probable_completion(l) &= \\
 &= \operatorname{argmax}_{(L,t)} = \operatorname{argmax} p(l, L, l_L, t) = \\
 &= p(t) p(L|t) p(l | L, t) p(l_L | l, L, t)
 \end{aligned} \quad (1)$$

The equation expresses the decomposition of the joint probability into a product of conditional probabilities. In order to compute the total probability, we can compute separately each factor in (1).

- $P(t)$, is an independent constant, which does not count in the computation of (1).
- $P(L | t)$, the probability that a last_name is in that piece of news, shows the connection between a topic and a last_name and may be computed in two ways:
 - by letting it take values in {60/100, 29/100, 10/100, 1/100} corresponding to “very frequent”, “frequent”, “little frequent” and “no clue” respectively.
 - By using the Bayes theorem, and keeping in mind that $p[t]$ does not count, we have $p(L | t) = p(t | L) p(L) p(t)$, which represents the probability that a certain last_name is in the newspapers, may be computed as a ratio. The probability $p(t | L)$ is normalized as above.

- $P(l | L, t)$ is the probability that a certain person is referred to by first_name in a piece of news. If nothing is known, we consider it 1/2. Otherwise, if there is any corpus based evidence available, we compute a yes/no function, considering whether it is more probable for that person to be called by first_name, and, consequently, a quantity is added/subtracted from 1/2.
- $P(l_L | l, L, t)$ is the probability that a person's name is made out of first_name and a last_name. We approximate it as {80/100, 20/100, 0} considering the probability that the last name is indeed a last_name. The "0" value means that we have unmatched abbreviations.

All the above probabilities have been calculated over the whole document collection.

The number of local coreferenced name mentions is large in the Adige500k corpus, over 200,000,. Most of these coreferences are between name mentions with the same value considered as strings. However, more than 12% of the total number of coreferences is ambiguous. The accurate resolution for these cases is essential for the extraction quasi-similar pairs.

3.3 Social Roles

By corefering the mentions denoting person within each piece of news we are also able to extract pairs of quasi-similar social roles. From Table 3 we know that we may expect that roughly one in third name mentions appears in a context where a word, or a phrase, denoting a social role is present. In order to maintain a good level of accuracy we considered only a limited number of all the possible contexts. We focused on the immediate context surrounding a name mention, considering the full noun phrases, which include the name mentions. The extent of full noun phrases is found by applying a simple stop word heuristics. The most successful simple heuristics seems to be the one guided by PoS and fixed phrases format.

In Figure 1 we plot the distribution of the social role pairs inside the news according to the person they refer to.

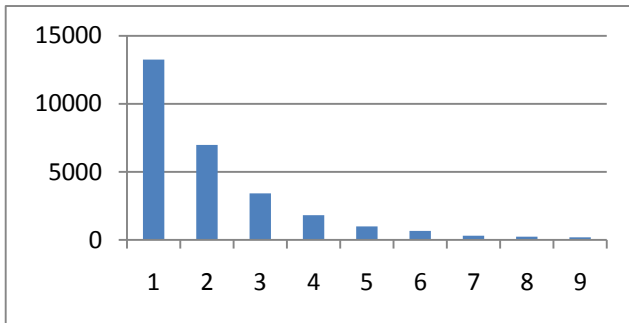


Figure 1. Social Roles per News

The social roles are found usually in front of name mentions, but in some cases they may follow them. The num-

bers are plotted in Figure 2, where the left bar indicates the preceding social roles.

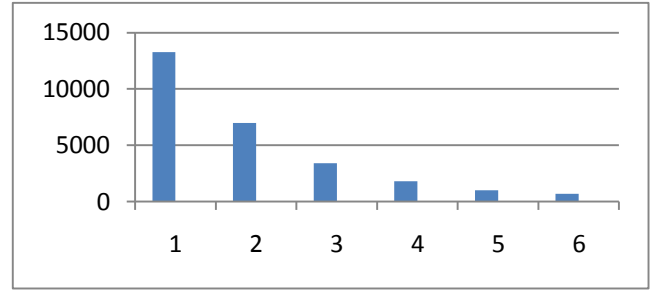


Figure 2. Social Roles Preceding/Following

In Table 4 we present the most social roles, together with their English translation present in Adige500k. The figures indicate how many social role words have been found by our extraction procedure, and they do not indicate the occurrences in the whole corpus.

Social Roles	occurrences
presidente [president]	78,907
assessore [counselor]	48,429
don [don]	23,902
consigliere [advisor]	19,470
direttore [principal]	19,044
professore[teacher]	16,950
avvocato [lawyer]	16,127
dottore [doctor]	15,766
segretario [assistant]	13,370
ministro [minister of state]	13,332

Table 4. Most frequent Social Roles

The social roles associated with coreferenced name mentions form pairs naturally. The pairs at this stage represent only candidate quasi-similar pairs. A quasi-similar pair has to guarantee that the elements are not associated by chance, or by mistake.

4 Quasi-Similar Pairs

The local coreference module extracts pairs of social roles found by coreferencing. In Table 5 we present an example of the candidate pairs. For each pair we record both the frequency of the respective pair and also the number of different pieces of news where the respective pair has been mentioned.

Candidate Pairs	#Occ	#News
albergatore assessore	5	3
allenatore commissario-tecnico	42	38
allenatore carabinieri	1	1
allenatore guida	2	2

Table 5. Candidates Pairs

We extract 78,118 pairs of candidates. The variance with respect to frequency and distribution in the corpus is big. In Figure 4 we present their distribution in the pieces of news.

Most of the pairs, almost 60%, appear only once and for the sake of having a better picture we do not plot them.

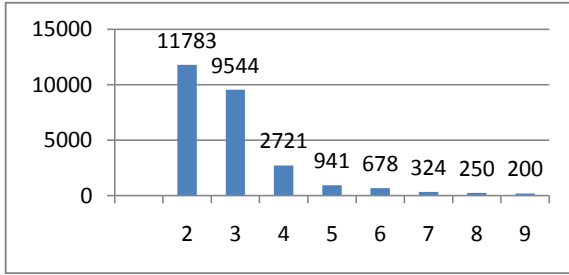


Figure 4. Candidate Pair Distribution

Not all these pairs are considered valid quasi-similar roles. We run a procedure that ensures that most of the cases representing the simple co-occurrences or mistakes are filtered out.

In Figure 5 we present the architecture of the system. Optionally one can start from the set of social roles present in SUMO. The figures we have reported in this section are without the SUMO initial set, because we want to analyze the performances of the system in the absence of any exterior resource.

There are two main conditions that the quasi-similar pairs must obey:

- each pair must have been observed with different persons
- the number of occurrences is above a predefined threshold for each pair.

We experimented only with few values, mostly because we were primarily interested in high accurate results. We choose two different set of constraints and consequently we obtained two different sets of quasi-similar pairs, say SET1 and SET2.

In order to evaluate the accuracy of the two sets we randomly choose a sample from each set, making sure that there are no duplicates. The cardinality of the sample for each set was the same, 125, even though the two sets had different cardinalities. The only criterion in choosing the sample size was statistics. We put together all the sample pairs and we randomly made a new two set sample, TSET1 and TSET2, each new set containing pairs from both SET1 and SET2. To these sets we add a sample of 75 pairs. Eventually each TSET set contained 200 pairs from both SET1 and SET2, with 75 pairs being common. The set TSET1 was assigned to a first human tester and the set TSET2 was assigned to a second human tester. In this way both human testers looked at pairs from SET1 as well as from SET2. The 75 pairs which TSET1 and TSET2 have in common, allow us to measure the inter-tester agreement.

The results are presented in Table 6.

Filtering Condition	#Quasi-Similar Pair	Accuracy
SET 1	8, 256	92%
SET 2	6, 230	97%

Table 6. Quasi-Similar Accuracy

The inter-tester agreement was very high, they agreed in 71 cases, which makes an 96% agreement. The null hypothesis that their accuracy varies with more than 5% can be rejected. In these conditions we can consider the results in Table 6 statistically significant.

In a separate experiment we considered only the social roles present in SUMO. We notice that it makes no difference in the final accuracy; therefore knowing already the social role words makes no difference in the end. However, with less restrictive filtering conditions it may turn out that it actually does.

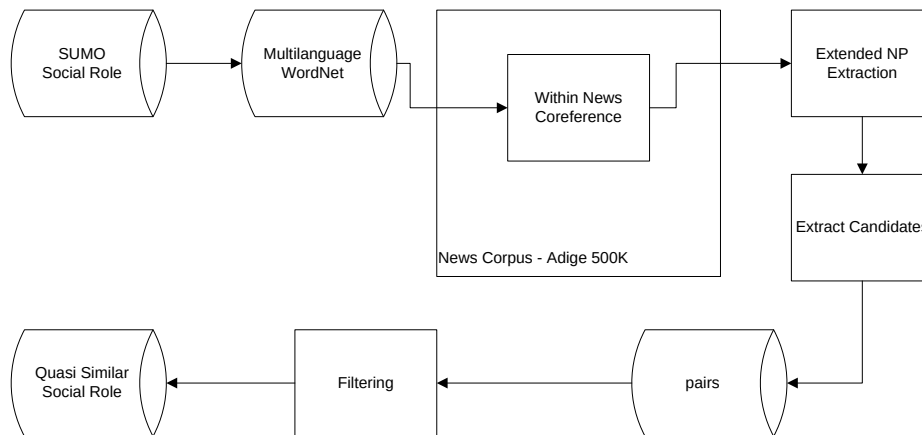


Figure 5. System Architecture

5 Conclusion and Further Work

We have presented a solution for learning quasi-similar pairs of words denoting social roles in specific contexts from a corpus. These pairs play an important role in textual entailment and in all related tasks because they represent specific information related to human knowledge which is not readily available in computable form.

The method we present is language independent, it requires only light computational tools and is very fast. The accuracy of the system is quite good.

In an independent evaluation experiment we concluded that the accuracy is between 92 and 97% percent depending on the restrictiveness of the filtering conditions.

The analysis carried on in the last section shows that the actual number of quasi-similar pairs is actually much bigger than the one we eventually considered after the filtering conditions. We believe that there are ways to design a filtering module which is more permissive without jeopardizing the accuracy. Our future research is oriented in this direction. Given the fact that quasi-similarity is a one to one relationship, it is possible to make a contradiction corpus for textual entailment, where all the counter examples are made by words that are not in a quasi-similar relationship. We intend to exploit this idea in the future.

References

- [Clark et al. 2007] Clark, P., Harrison, P., Thompson, J., Murray, W., Hobbs, J., and Fellbaum, On the role of lexical and world knowledge in RTE3. PASCAL, 2007
- [Cohn&Lapata 2009] Cohn, T., & Lapata, M.. Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*, 34(1), 637–674, 2009
- [Dagan, Glickman, & Magnini, 2006] Dagan, I., Glickman, O., & Magnini, B. The PASCAL recognising textual entailment challenge. *Machine Learning Challenges. Lecture Notes in Computer Science*, Vol. 3944, pp. 177–190. Springer-Verlag. 2006.
- [Dagan et al. 2008] Dagan, I., Dolan, B., Magnini, B., & Roth, D. Recognizing textual entailment : Rational, evaluation and approaches. *Nat. Lang. Engineering*, 15(4), i–xvii. Editorial of the special issue on Textual Entailment. 2008
- [Deleger & Zweigenbaum, 2009] Deleger & Zweigenbaum , Extracting lay paraphrases of specialized expressions from monolingual comparable medical corpora. In *Proc. of the 2nd Workshop on Building and Using Comparable Corpora: from Parallel to Non-parallel Corpora*, pp. 2–10, Singapore, 2009
- [Elhadad&Sutaria 2007] Elhadad&Sutaria Mining a lexicon of technical terms and lay equivalents. In *Proc. of the Workshop on BioNLP*, pp. 49–56, Prague, Czech Republic., 2007
- [Giampiccolo 2008] Giampiccolo, D., Dang, H., Magnini, B., Dagan, I., & Dolan, B. The fourth PASCAL recognizing textual entailment challenge. In *Proc. of the Text Analysis Conference*, pp. 1–9, 2008
- [Hovy2003] Hovy E., Text summarization. In Mitkov, R. (Ed.), *The Oxford Handbook of Comp. Linguistics*, chap. 32, pp. 583–598. Oxford University Press. 2003 [Jacquemin&Bourigault, 2003]
- [Hovy et. al 2007] Bhagat, R., P. Pantel and E.H. Hovy. LEDIR: An Unsupervised Algorithm for Learning Directionality of Inference Rules. Poster in *Proceedings of the EMNLP conference*. Prague, 2007.
- [Kouleykov et. al. 2005] Kouleykov, M., and Magnini, B.. Tree edit distance for textual entailment. In *Proceedings of RALNP-2005, International Conference on Recent Advances in Natural Language Processing*, Borovets, Bulgaria, pp. 271–8. 2005
- [Lin&Pantel 2001] Lin, D., & Pantel, P. Discovery of inference rules for question answering. *Nat. Lang. Engineering*, 7, 343–360. 2001
- [Magnini et. al 2006]. Magnini, B., Pianta, E., Girardi, C., Negri, M., Romano, L., Speranza, M., Bartalesi Lenzi V., Sprugnoli, R. I-CAB: the Italian Content Annotation Bank, In *Proceedings of LREC-2006*, Genova, Italy, 2006
- [Niles 2003] Niles, I., and Pease, A.. Towards a Standard Upper Ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001)*, Chris Welty and Barry Smith, eds, Ogunquit, Maine, October 17-19, 2001.
- [Pianta et. al. 2002] Emanuele Pianta, Luisa Bentivogli, Christian Girardi. MultiWordNet: developing an aligned multilingual database" In *Proceedings of the First International Conference on Global WordNet*, Mysore, India, January 21-25, 2002.
- [Pianta et. al. 2006] Magnini B., E. Pianta, O. Popescu and M. Speranza (2006). Ontology Population from Textual Mentions: Task Definition and Benchmark. In *Proceedings of the OLP2 workshop on Ontology Population and Learning*, Sidney, Australia, 2006. Joint with ACL/Coling
- [Popescu&Magnini 2007] Popescu O. Magnini B., Octavian Popescu, Bernardo Magnini: "Inferring Coreferences among Person Names in a Large Corpus of News Collection". In *Lecture Notes in Computer Science*. Volume 4733: AI*IA 2007: Springer Berlin/Heidelberg. 2007
- [Szpektor et. al 2008] Szpektor, I., & Dagan, I. Learning entailment rules for unary templates. In *Proc. of the 22nd Int. Conf. on Comp. Linguistics*, pp. 849–856, Manchester, UK., 2008
- [Voorhes 2001] Voorhees, E. The TREC QA track. *Nat. Lang. Engineering*, 7(4), 361–378., 2001

Bootstrapping Relation Extraction Using a Parallel Corpus

Michael Glass

Department of Computer Science
University of Texas at Austin
mrglass@cs.utexas.edu

Ken Barker

Department of Computer Science
University of Texas at Austin
kbarker@cs.utexas.edu

Abstract

Relation extraction is the task of finding entities in text connected by semantic relations. Bootstrapping approaches to relation extraction have gained considerable attention in recent years. These approaches are built with an underlying assumption, that when a pair of words is known to be related in a specific way, sentences containing those words are likely to express that relationship. Therefore, sentences containing the pair of words may be used as training data for the relation extractor. We test this assumption for various relations drawn from two domains, using parallel and non-parallel corpora. We find that the assumption holds with substantially greater probability for parallel corpora.

1 Background and Related Work

Due to the enormous variety of expression in natural language, semantic structures can surface in many ways. Finding each of these ways manually, either by annotating or constructing interpretation rules by hand is impractical. One goal of DARPA's Machine Reading Project is to minimize this human effort when moving to new domains. Semi-supervised bootstrapping methods in relation extraction are a natural choice because of their ability to benefit from a small amount of manual effort and a large, unannotated corpus.

Algorithm 1 gives a general outline of relation extraction bootstrapping. This algorithm bootstraps a single relation, although in some cases it may be beneficial to bootstrap multiple relations together [Carlson *et al.*, 2009]. It begins with a corpus, C , and a seed set of related pairs of instances, R , for example $\{(Cavaliers, 109), (Suns, 91) \dots\}$ for the *team-Score* relation. Alternatively, we may obtain seeds from a high precision (but low recall) relation extractor, in which case we first run the relation extractor over the corpus to obtain R . The first step of the bootstrapping loop is to label all occurrences of the pairs from R in C . Then an extractor is trained to identify the labeled sentences as examples of the target relation. The process repeats with the trained extractor providing an expanded set R .

In a seminal work, Hearst [1992] demonstrated a means of finding hyponyms from text using an initial set of simple syntactic patterns and a semi-automatically discovered set of ad-

Algorithm 1 A general relation bootstrapping algorithm

BOOTSTRAP(R, C)

```
1  $E \leftarrow \emptyset$ 
2 repeat  $O \leftarrow \text{LABELOCCURRENCES}(R, C)$ 
3        $E \leftarrow \text{TRAINEXTRACTOR}(O)$ 
4        $R_{orig} \leftarrow R$ 
5        $R \leftarrow \text{RUNEXTRACTOR}(E, C)$ 
6 until  $R = R_{orig}$ 
7 return  $E$ 
```

ditional syntactic patterns. Many researchers have also used parallel (or comparable) corpora to find paraphrases [Barzilay and Lee, 2003]. Shinyama *et al.* [2002] used parallel news stories to extract dependency paths with similar meanings based on shared named entity fillers. A related idea, distant supervision [Mintz *et al.*, 2009], uses a database of known facts to label a separate corpus of natural language.

Algorithm 1 relies on the implicit assumption that when we encounter a sentence containing a pair of words known to be in a specific semantic relation, that sentence is likely to express that relation. This assumption has been at least partially validated by the success of the NLP bootstrapping methods that depend on it. In this work, we will measure the degree to which it holds for various relations in parallel and non-parallel corpora.

2 Automatically Constructed Parallel Corpora

The Machine Reading Project motivated relation extraction on two domains: National Football League (NFL) and Intelligence Community (IC). The relations in the NFL ontology were general enough to apply to any team sport, so we will often refer to this as the Sports domain.

The abundance of news articles reporting on the same event makes parallel or comparable newswire corpora an attractive resource. Dolan *et al.* built an aligned corpus [2004], useful in paraphrase acquisition research, from news articles.

We built a large parallel corpus for both the Sports domain and the IC domain from clustered newswire. We used Google News¹ to locate and cluster articles describing the same story.

¹<http://news.google.com/>

Each document cluster covers a single news story, often a single event such as a game or terrorist attack. By searching Google News with domain relevant keywords, a set of results similar to Figure 1 is retrieved. By following the link to “all 285 articles” a set of documents all describing the same event can be gathered. Because both the relevance to the cluster and the quality (pagerank) of the articles decline with the number of search results retrieved, we limited the number of articles to the top ranked one third, or at most 100.



Figure 1: Example of a Google News article cluster

Each corpus is approximately half a gigabyte of text, gathered over about four months. On each day the corpus gathering software downloaded the clusters for the top ranked stories for each search term. The sports corpus was constructed by querying for news articles with the keywords: “NFL”, “NBA”, “NCAA football”, and “MLB”. The IC corpus was constructed from the keywords: “intelligence community”, “conflict region”, “Al Qaeda”, “Taliban”, “foreign election”, and “insurgent”. The sports corpus contains 145,000 documents across 3861 clusters, with an average of 37.5 documents per cluster. The IC corpus is similar, with 130,000 documents, 3114 clusters and 41.5 documents per cluster.

The news articles were automatically cleaned of the boilerplate typical of web articles and stripped of all HTML. The resulting text documents were segmented into sentences using Stanford’s [Klein and Manning, 2003] sentence segmenter.

Because of the way the corpus was gathered, we know only that the documents inside each cluster were considered to be about the same story by Google News. We do not know that documents in other clusters are not about the same story. Often, multiple consecutive days will have articles about the same event. Given our architecture, these would appear in different clusters.

3 Experimental Setup

To determine the potential value of a parallel corpus for bootstrapping relation extractors, the experiment measures how consistently a relation holds between two seed instances within related news stories and across unrelated stories. Table 1 shows the five relations from each domain we considered. The gloss is an English explanation of the relation, where x and y are the two related entities. The search pattern is a snippet of text we used to locate sentences in the corpus containing entities in the relationship of interest.

Using the search patterns we first gathered sentences that potentially expressed the relation. From these we manually located sentences that actually expressed the target relation and located the words that were related. We refer to the pairs

of words in these sentences as the seed instances. These pairs would be the initial set R in algorithm 1.

Because all sentences in which the seed instances are syntactically related will be labeled as occurrences of the target relation in the next bootstrapping step, it is essential that the instances be suitably specific. If the seed instances are overly general, such as *(group, he)* for *isLedBy*, the LABELOCCURRENCES process will have low precision. If the seed instances are too specific, such as *(The surging Los Angeles Lakers, Monday's game)* for *teamInGame*, very few sentences will be labeled as occurrences of the relation. To balance these considerations, we adopted the following seed instance selection strategy. First, no pronouns were accepted. For teams, either the name or city was acceptable. For people, a last name or descriptive general noun like “defendant” was accepted. For organizations referred to by name, only the most descriptive portion of the name was retained, for example “Harvard” for “Harvard University”.

To simulate the LABELOCCURRENCES function in algorithm 1, a simple extractor randomly selected additional sentences in the corpus that contained a pair of seed instances. For each seed instance pair, it selected ten sentences to evaluate the precision of LABELOCCURRENCES within a single document cluster, called the “inside cluster” sentences. Since there might not be a single document cluster that had ten occurrences of a seed instance pair, inside cluster sentences were added when there were at least two within a single cluster: the extractor continued randomly adding all matching sentences from suitable clusters until the number of sentences minus the number of document clusters they were drawn from equaled or exceeded ten. We also selected another ten sentences spread across document clusters to test the precision of LABELOCCURRENCES without document clustering, called the “outside cluster” sentences.

We required the sentences found in this way to have a close syntactic relationship between the two seed instances. We measured this by parsing the sentences and rejecting any sentence where the words were separated by a dependency path of length greater than five.

4 Evaluation

To evaluate the precision of the LABELOCCURRENCES function, with and without document clustering, we used workers on Amazon’s Mechanical Turk² service. Other researchers [Snow *et al.*, 2008] have shown that for some simple tasks in natural language processing, including textual entailment, Mechanical Turk non-experts can provide annotations with the same level of quality as expert-built gold standards.

For each relation a gloss was written with its meaning expressed in plain English (see Table 1). The found sentences, along with the filled gloss for the relation’s meaning were the Human Intelligence Tests (HITs). Each HIT (the labeled occurrence) was given to three workers, to test agreement. The workers were asked to mark the sentence-summary pair as correct or incorrect (and if incorrect, select a reason why).

²<http://mturk.com>

Relation	Gloss	Search Pattern
NFL (Sports) Relations		
gameDate	The game referred to as ' <i>x</i> ' was played on <i>y</i>	" on "
gameWinner	The team, <i>x</i> , won the game referred to as ' <i>y</i> '	" won "
gameLoser	The team, <i>x</i> , lost the game referred to as ' <i>y</i> '	" lost "
teamInGame	The team, <i>x</i> , played in the game referred to as ' <i>y</i> '	" between "
teamScore	The team, <i>x</i> , scored <i>y</i> points	" scored "
Intelligence Community Relations		
eventLocationGPE	The event, <i>x</i> , took place in <i>y</i>	" in "
hasCitizenship	The person, <i>x</i> , is a citizen of <i>y</i>	" citizen of "
injuringHumanAgent	The group, <i>x</i> , was responsible for the attack, <i>y</i>	" claimed responsibility "
isLedBy	The organization, <i>x</i> , is led by <i>y</i>	" headed by "

Table 1: Relations for filler extraction

We provided example sentence-summary pairs to illustrate the three different types of incorrect summaries.

- Determine if the sentence states or strongly implies the partial summary.
- If the sentence does not actually state (or strongly imply) the relationship, it is incorrect, even if you know the partial summary is true.
- If it is incorrect, consider why it is not correct.
 - Is a word in bold not of the type stated?
 - * Sentence: John **won** an award in **New York**.
 - * Summary: The team, **New York**, won the game referred to as **won**.
 - * Explanation: "New York" does not refer to a team and "won" does not refer to a game.
 - Is there no direct relationship between the words?
 - * Sentence: Kobe Bryant of the Los Angeles **Lakers** will **play** in the All-Star Game.
 - * Summary: The team, **Lakers**, played in the game referred to as **play**.
 - * Explanation: There is only an indirect relationship between the Lakers and the playing event.
 - Is the relationship between the words different from the summary?
 - * Sentence: President **Obama** spoke at **Harvard** today.
 - * Summary: The person, **Obama**, attended the school, **Harvard**.
 - * Explanation: The sentence says that Obama spoke at Harvard, not attended
- Tense (past, present, future) is unimportant.

5 Results

The Fleiss' Kappa for Mechanical Turk worker agreement across all relations was 0.333 indicating "Fair agreement" [Landis and Koch, 1977]. The *gameWinner* and *gameLoser* relations had low precision (< 20%) and low agreement (< 0.1) for both the inside and outside cluster sentences. We exclude these relations from further analysis.

Mechanical Turk workers were unanimous in about half of the HITs. For the others, we reviewed a sample of 25 HITs. Of those, we agreed with the majority of workers 19 times. In cases where workers disagreed, they often had legitimate reasons for their disagreements. Consider the sentence below

along with the gloss of the hypothesized relation.

Sistan-Baluchistan province a major transit route for narcotics has been hit by a string of attacks and kidnappings, that authorities blame on Jundollah.

The group, **Jundollah**, was responsible for the attack, **attack**

Answers	correct	1	incorrect	2
---------	---------	---	-----------	---

The workers were instructed to mark the partial summary correct if it was stated or "strongly implied" by the sentence. Whether the blame of authorities is enough to strongly imply responsibility is obviously a subjective matter.

5.1 Effect of Parallel Corpus

To measure the inside cluster precision for a relation, we located all sentence-summary pairs where the workers unanimously agreed the partial summary was correct. Then we tallied the majority opinion - whether correct or incorrect - on every other sentence-summary pair from the same cluster. To measure outside cluster precision we simply gathered the majority opinion for every outside cluster sentence. Precision is then $correct / (correct + incorrect)$. The ratio reported in Figure 2 is the outside cluster precision divided by the inside cluster precision.

The difference between the inside and outside cluster precision is significant, even adjusting for multiple testing, for all relations but the last three: *injuringHumanAgent*, *isLedBy*, and *teamInGame*.

6 Conclusions

Figure 2 shows the importance of using a parallel corpus for at least some relations. Overall the inside cluster precision was 0.70 and the outside cluster precision was 0.39, suggesting a parallel corpus may lead to an error rate reduction of 50% in LABELOCCURRENCES. The effect varied considerably based on relation.

We expected a minimal impact for the *gameDate*, *teamInGame* and *injuringHumanAgent* relations and a substantial impact for the others. We reasoned that when the typical instance pairs for these relations are present together in a sentence it is rare that they are related in anyway except the target relation. Although we were unable to test some relations effectively, with the exceptions of *gameDate* and *isLedBy* our hypothesis was validated.

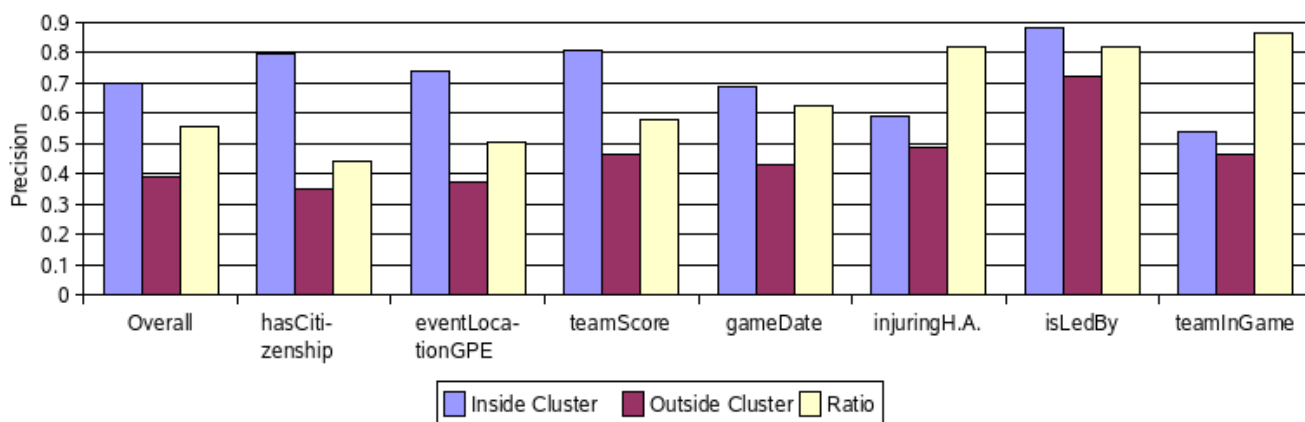


Figure 2: Impact of a parallel corpus on instance relationship precision

The dramatic drop from inside cluster to outside cluster precision for the *hasCitizenship* relation can be explained by the fact that, not coincidentally, a citizen of a nation is typically related to that nation in many ways. For *teamScore*, we see that within a cluster (usually about a single game) if the relationship holds between a team and a score in one sentence in that cluster, it is very likely to hold between all other syntactically related occurrences of those same instances. However, outside the cluster it is less than 50% likely that the relation holds. Often the score will actually be the *other* team's score.

For many cases the assumption of a single relation between seed instance pairs does not hold generally, but is reliable within a cluster of documents on a single story. Bootstrapping approaches to relation extraction can take advantage of this result by using parallel corpora for relations that can benefit.

Acknowledgments

The authors gratefully acknowledge the support of Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0172. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, AFRL, or the US government.

References

- [Barzilay and Lee, 2003] Regina Barzilay and Lillian Lee. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *NAACL '03: Proc. of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 16–23. Association for Computational Linguistics, 2003.
- [Carlson et al., 2009] Andrew Carlson, Justin Betteridge, Estevam R. Hruschka, Jr., and Tom M. Mitchell. Coupling semi-supervised learning of categories and relations. In *Proc. of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing*, SemiSupLearn '09, pages 1–9. Association for Computational Linguistics, 2009.
- [Dolan et al., 2004] Bill Dolan, Chris Quirk, and Chris Brockett. Unsupervised construction of large paraphrase corpora: exploiting massively parallel news sources. In *COLING '04: Proc. of the 20th International Conference on Computational Linguistics*, page 350. Association for Computational Linguistics, 2004.
- [Hearst, 1992] Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proc. of the 14th Conference on Computational Linguistics*, pages 539–545. Association for Computational Linguistics, 1992.
- [Klein and Manning, 2003] Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *ACL '03: Proc. of the 41st Annual Meeting on Association for Computational Linguistics*, pages 423–430. Association for Computational Linguistics, 2003.
- [Landis and Koch, 1977] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, March 1977.
- [Mintz et al., 2009] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proc. of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2, ACL '09*, pages 1003–1011. Association for Computational Linguistics, 2009.
- [Shinyama et al., 2002] Yusuke Shinyama, Satoshi Sekine, and Kiyoshi Sudo. Automatic paraphrase acquisition from news articles. In *Proc. of the Second International Conference on Human Language Technology Research*, pages 313–318. Morgan Kaufmann Publishers Inc., 2002.
- [Snow et al., 2008] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *EMNLP '08: Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 254–263. Association for Computational Linguistics, 2008.

Improving Recall on Conjunctive Queries against Text Documents using Query-Driven Hypothesis Generation

Ken Barker

University of Texas at Austin
Austin, Texas, USA
kbarker@cs.utexas.edu

James Fan

IBM Watson Research Center
Hawthorne, New York
fanj@us.ibm.com

Chris Welty

IBM Watson Research Center
Hawthorne, New York
welty@us.ibm.com

Abstract

We are experimenting with a task that involves evaluating conjunctive queries over subject-property-object graphs that are generated from the results of processing text. Given the fact that NLP results are imperfect, errors in individual query conjuncts multiply, often causing recall to drop dramatically as queries add terms. To address this, we present a hypothesis generation technique based on identifying missing graph edges representing type or other binary relations that were not extracted from the source text, and show that with suitable hypothesis validation techniques drawn from the literature and the Watson Q/A system, we significantly improve recall of conjunctive queries while still improving F-measure.

Topics

Query Analysis; Bridging knowledge gaps in NLP

1 Introduction

As information extraction-based question answering systems are presented with more complex queries, low recall becomes the dominating factor in performance. In particular, for queries that are a conjunction of terms, the overall recall can be severely degraded by poor term recall. In the worst case, query recall can be zero even for non-zero term recall if the variable bindings over individual terms do not overlap. For example, consider the conjunctive query to find Jordanian citizens who are members of Hezbollah:

```
citizenOf(?P, Jordan) ^  
hasMember(Hezbollah, ?P)
```

Even if the recall of each term is non-zero, their bindings for the variable `?P` must overlap in order for the query recall to be non-zero.

In this paper we describe a system that uses solutions to subsets of a conjunctive query as candidate solutions to the full query, then attempts to confirm the candidate solutions using various kinds of inference and external resources.

1.1 Query Answering Scenario

Our query answering task is the current evaluation scenario for DARPA's Machine Reading program. We are given an ontology of types (unary predicates) and binary relations, a target corpus of 1000 documents and 50 questions. The questions are a conjunction of terms from the ontology with some arguments variable and some already bound.

For example, consider the query asking to find all terrorist groups that were agents of bombings in Lebanon on October 23, 1983:

```
Select ?T where  
TerroristOrganization(?T) ^ Bombing(?B) ^  
mediatingAgent(?B, ?T) ^  
eventLocation(?B, Lebanon) ^  
eventDate(?B, 1983-10-23) .
```

The query answering system must find all bindings for the variable `?T` that satisfy the conjunctive query and also report where in the target corpus the answers are found. In practice, this means finding spans of text ("provenance") expressing the relations in the query.

Our system consists of two main parts: an "NLP stack" and a Query Answering module. The NLP Stack contains parsers, coreference resolution modules and entity and relation extractors trained on the ontology and text in the domain. Its main component is a state-of-the-art information extraction system developed at IBM primarily for the ACE evaluation, and adapted to the specific ontology of this task [Florian *et al.*, 2004]. The NLP Stack is run over the target corpus to produce an "extraction graph", which includes entities and relations between them, as well as pointers to their mentions in the text. The graph also associates confidence measures on extracted entities and relations.

2 Observations on a Dry Run

Prior to the official evaluation task, we were given a "dry run" evaluation with an almost identical ontology, a different corpus of 1,000 documents and 50 different questions. Since our information extraction component was not yet trained on the domain, we used a simple bootstrapping extractor [Glass *et al.*, 2009] in the NLP stack. Although the

dry run task was intended primarily to shake out bugs in the process, and although our NLP Stack was much more primitive than we knew it would ultimately be, our system’s performance was nonetheless cause for concern. We were able to produce answers to only eight of the 50 dry run queries. For queries with multiple relations we provided no correct answers at all. For queries consisting of one relation and type restrictions, we scored an F1 of .27. This result suggested that we couldn’t even begin to consider how to improve precision until we found a way to improve recall.

2.1 Query Term Ablation Experiments

In light of the poor dry run results, we decided to investigate individual query conjuncts to see if improving the NLP Stack’s recall on certain problematic types or relations would improve QA recall. We found that although there were only a small number of types and relations with zero extractions, there were still many questions for which we were producing no answers even when the individual query terms all had recall well above zero.

To investigate the source of errors, we re-ran QA ablating various conjunct types and combinations and found (unsurprisingly) that the weakened queries produced more answers (Table 1). Overall, with various combinations of up to two ablations we doubled the number of queries for which we could produce answers to 16 (no single row in Table 1 adds all eight new queries). Note that producing an *answer* to a query does not mean it is a correct one; we did, however, observe correct answers among the relaxed query result sets.

Relaxing queries by deleting terms gives the potential to improve recall, but changes the semantics of the queries (in some cases, rendering them vacuous). Our goal, then, was to find a way to take advantage of term-based query relaxation to increase recall, without changing query semantics and without ruining precision.

3 Query-Driven Hypothesis Generation

3.1 Generating Hypotheses

Our solution was to relax queries by removing query terms, find solutions to the remaining set of terms and use those solutions to ground the relaxed queries for further analysis. In a sense we are looking for missing links in a graph that, if added, would result in a new query solution. The variable bindings for the relaxed query results give us a place to start (see Figure 1).

Each solution to a query Q is a single binding over the set of variables V shared among N query terms. We relax Q by removing a query term, leaving a relaxed query Q' having some subset of variables V' shared among the remaining $N-1$ terms. The solutions to Q' provide bindings over V' . We use those bindings to ground all of the original N query terms in Q . These terms of Q grounded using the bindings from solutions to Q' are hypotheses to be tested. Of course, $N-1$ of the hypotheses are already satisfied by the extractions, but since extractions are potentially noisy, we subject them to hypothesis testing, too.

<i>terms ablated</i>	<i>queries with answers</i>	<i>total answers</i>
<i>no ablations</i>	8	266
count terms	8	266
datespec	8	266
all date terms	9	272
event locations	9	862
entity types	9	1168
count+datespec	8	266
count+alldate	9	272
count+evloc	11	2506
count+type	9	1168
datespec+evloc	11	870
datespec+type	9	1168
alldate+evloc	12	1000
alldate+type	10	1240
evloc+type	11	13715

Table 1. Number of answerable queries and number of answers when ablating categories of query terms.

Each set of N query terms is a hypothesis set: if the hypotheses in a hypothesis set can be proven, the hypothesis set constitutes a new solution to Q . If no solutions to subqueries of size $N-1$ are found, we consider subqueries of size $N-2$, and so on down to subqueries of size $N/2$. (In practice we rarely go lower than $N-2$: the number of hypotheses begins to explode, and they are rarely related enough to be useful).

For example, consider again the query from section 1 asking to find all terrorist groups that were agents of bombings in Lebanon on October 23, 1983:

```
Select ?T where
1. TerroristOrganization(?T) ^
2. Bombing(?B) ^
3. mediatingAgent(?B,?T) ^
4. eventLocation(?B,Lebanon) ^
5. eventDate(?B,1983-10-23).
```

The system would consider five relaxations at level 1, corresponding to the subqueries of length four resulting from ablating each term in turn:

Q: $R1(x,y), R2(x,z), R3(z,w)$

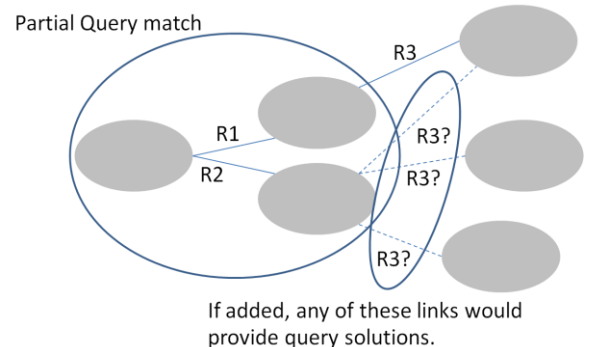


Figure 1. Hypotheses are formed from links that, if added to a graph, would provide new query solutions.

1. find all bombings in Lebanon on 1983-10-23 with agents (hypothesize that the agents are terrorist organizations)
2. find all events in Lebanon on 1983-10-23 by terrorist orgs (hypothesize that the events are bombings)
3. find all bombings in Lebanon on 1983-10-23 (all known terrorist orgs are hypothetical agents)
4. find all bombings by terrorist orgs on 1983-10-23 (hypothesize that the bombings were in Lebanon)
5. find all bombings by terrorist organizations in Lebanon (hypothesize that the bombings were on 1983-10-23)

If no solutions for any of the five subqueries are found, the system will attempt to generate hypotheses for the twenty subqueries of length three, and so on.

Two specific cases in hypothesis generation are worth describing in more detail. First, it is often the case that a variable appears in exactly two terms in a query: a type term and a relation term (for example, variable $?T$ above). Removing the *type* term produces a subquery in which the corresponding argument of the relation term is more weakly constrained. Relaxation 1 above illustrates this case: find all bombings in Lebanon on 1983-10-23 by any agent, not just terrorist organizations. The solutions to the relaxed subquery provides a list of candidate bindings of the variable $?T$ consisting of all agents of appropriate bombings. The resulting hypotheses are that those agents are of type `TerroristOrganization`. But removing the *relation* term produces a much weaker subquery, that of relaxation 3. Since $?T$ is a singleton variable in the subquery, *all* instances of type `TerroristOrganization` are candidate agents of the bombing. Such subqueries typically produce a large number of “shot-in-the-dark” hypotheses, which sometimes require careful handling during hypothesis checking (see section 4).

The second special case is when subquery solutions do not bind a variable. This occurs for subqueries of length $N-k$ for any variables that appear k (or fewer) times in the original query. For example, if the above query produced no solutions to subqueries of length $N-1$, the system would consider subqueries of length $N-2$. At this level, we would produce a subquery where both terms in which $?T$ appears are removed. That is, we would produce a subquery in which $?T$ does not appear. Solutions to the subquery would provide candidate bindings for $?B$, which would generate *mediatingAgent* hypotheses in which the first argument is bound, but the second is variable. For now, we filter out hypotheses with unbound arguments, though our hypothesis checking architecture could certainly admit checkers capable of dealing with partially bound hypotheses.

3.2 Query Optimization

The one-argument *type* terms and two-argument *binary relation* terms making up conjunctive queries are drawn from a given, domain-specific ontology. The queries in the evaluation are not optimized. Since our approach finds all solutions from the extraction graph over all query subsets of a given length, optimizing the query is important.

Solutions to subqueries are variable bindings from extracted facts (the direct result of running the NLP stack on a

fixed corpus of 1000 documents) and inferred facts that result from the application of a small, arbitrary rule base that allows taxonomic reasoning as well as some domain knowledge (see section 4.2). Prior to hypothesis generation, we cache the number of extracted and inferred groundings of each type and relation term. When a query is posed, we first sort the query terms from least frequent to most frequent. Physically reordering the query terms not only optimizes the query, but guarantees that all subqueries considered in hypothesis generation are also optimized.

Another advantage to optimizing the queries in this way is that subqueries that generate too many solutions can simply be truncated by removing the last (most productive) term(s). In practice, this doesn’t usually change the semantics of the query, since the most productive terms are usually type terms constraining the arguments of relation terms. The arguments of relation terms are usually already constrained by the ontology. Truncation does reduce the number of hypotheses generated for a subquery, but the lost hypotheses are seldom useful (for example, find all instances of `Person` and hypothesize that they were born in Japan).

4 Hypothesis Checking

4.1 Architecture

Each hypothesis set contains hypotheses in the form of ground relation and type terms, which, if supported, provide a new answer to the original query. All hypotheses are passed to a stack of hypothesis checkers. Each hypothesis checker reports its confidence that the hypothesis holds and gives a pointer to a span of text in the target corpus that supports the hypothesis (the *provenance*). In the current architecture, hypothesis checkers may also choose whether to allow a hypothesis to flow through to the next checker.

A significant property of the architecture is that an individual hypothesis checker can be any arbitrary module capable of reporting support for a hypothesis with confidence and provenance, including external IR systems, IE systems, knowledge bases, reasoners or even independent question answering systems.

4.2 Hypothesis Checkers

Extraction Graph

The original extraction graph is itself just one of the hypothesis checkers in our stack. By default, this means that the hypotheses corresponding to the $N-1$ terms *not* ablated are guaranteed to be supported, with the original extraction confidence and provenance. By including these terms as hypotheses and rechecking them, our system could adjust their confidence using other hypothesis checkers and even reject them if their confidence is low. Other extraction graphs could also be plugged in as checkers (extraction graphs over other corpora, or even a new extraction graph over the same corpus with different NLP stack configurations).

Knowledge Base

A second hypothesis checker is a knowledge base allowing taxonomic inference as well as more complex rules.

For our evaluation, the knowledge base contains mostly rules derived directly from the supplied domain ontology. It also contains a small number of general, domain-independent rules (such as family relationships) and some geographic knowledge.

Rules derived directly from the ontology include simple superclass-subclass rules ($\text{Bombing}(\text{?x}) \rightarrow \text{Attack}(\text{?x})$), simple relation-subrelation rules ($\text{hasSon}(\text{?x}, \text{?y}) \rightarrow \text{hasChild}(\text{?x}, \text{?y})$) and simple relation inverse rules ($\text{hasChild}(\text{?x}, \text{?y}) \rightarrow \text{hasParent}(\text{?y}, \text{?x})$). We also derived about 40 more complex rules automatically from the ontology, based on specialization of the domain or range of subrelations ($\text{hasSubGroup}(\text{?x}, \text{?y}) \wedge \text{HumanOrganization}(\text{?x}) \rightarrow \text{hasSubOrganization}(\text{?x}, \text{?y})$).

Finally, the knowledge base contains some domain-independent rules from the Component Library [Barker *et al.*, 2001] such as spatial and family relationships ($\text{hasChild}(\text{?x}, \text{?y}) \wedge \text{hasGender}(\text{?y}, \text{male}) \rightarrow \text{hasSon}(\text{?x}, \text{?y})$) as well as geographic subregion and demonym rules from WordNet [Miller, 1990].¹

Using the knowledge base as a hypothesis checker allows us to confirm, for example, the hypothesis

`eventLocation(?B, Lebanon)`

when the extraction graph contains

`eventLocation(?B, Beirut).`

Since all the rules are deductive, we made the confidence in the inferred consequent simply the product of the confidences of the antecedents with no further penalty. We have also begun to experiment with abductive application of rules with a heavier confidence penalty.

Watson

Whereas the extraction graph and knowledge base can check both relation hypotheses and type hypotheses, checkers may provide support for only one or the other.

Watson [Ferrucci *et al.*, 2010] is an open domain question answering system that has successfully beaten the best human contestants on America’s most popular quiz show *Jeopardy!*. It takes a question expressed in English, generates candidate answers by searching through its corpora including a variety of encyclopedias, dictionaries, news articles, etc. The candidates are then scored by a large number of different scoring algorithms, such as TyCor (see below), and passage matching. The scores for each candidate are combined into a single confidence to indicate the estimated likelihood of that candidate being correct.

In order to use Watson in our checker stack, we had to generate English glosses of the hypotheses in the form that Watson expects. Returning to our running example, assume that there are solutions to subquery 3: the NLP stack was able to extract instances of bombings in Lebanon on 1983-10-23, binding `?B` one or more times, but was unable to extract any `mediatingAgent` relations for those bombings. The hypotheses will be ground instances of `mediatingA-`

`gent(?B, ?T)` for all bindings of `?B` extracted and all instances of `?T` satisfying `TerroristOrganization(?T)` – that is, *all* of the instances of `TerroristOrganization` extracted. We generate English glosses of each hypothesized grounding of `mediatingAgent`. For example, if the bindings for `?T` included Hamas, Hezbollah and Islamic Jihad, we would generate the following Watson question:

What terrorist organization was responsible for the October 23 1983 bombing in Lebanon?

and these candidate answers: Hamas, Hezbollah, Islamic Jihad. Watson generates confidences for each of these answers based on all of its resources. Since answers to questions in our evaluation must come from the target corpus, we restricted Watson to finding supporting passages from that corpus only.

TyCor Type Checking

TyCor is a component in Watson for determining whether a candidate answer is of a particular type, loosely based on the notion of *type coercion* [Pustejovsky, 1995].

TyCor takes a candidate answer string and a lexical answer type (LAT) string, and generates a score indicating whether the candidate answer string can be an instance of the LAT string using a four step process. First, it maps the candidate answer string to an *instance* in some source of typing information (see below). Second, it extracts the *type* of the candidate answer instance. Third, it maps the given LAT string to a *type* in the same source, and finally, it checks to see if the candidate answer’s type matches the LAT’s. TyCor outputs a confidence between 0-1 that the candidate answer is an instance of the LAT; a confidence of 1 indicates certainty, 0 indicates no evidence was found (but does not indicate that the instance is not of the type). TyCor uses and combines results from a variety of sources such as Yago [Suchanek *et al.*, 2007], Wikipedia categories, WordNet [Miller, 1990], VerbNet [Kipper-Schuler, 2005], corpus aggregate statistics, etc. These sources are not mutually aligned, rather the candidate and LAT mappings are implemented independently for each, and the reliability of each source for the task is learned through large amounts of training data. The version of TyCor used for this experiment had been trained for the *Jeopardy* QA task.

To use TyCor, we convert type hypotheses to pairs of strings corresponding to the binding of the variable in the type term and the type itself. For example, the hypothesis `Bombing(E781)`, where E781 is an entity extracted from the corpus, would generate the TyCor pair: (“violent explosions”, “bombing”), where “violent explosions” is the mention text for E781. TyCor would return a confidence value between 0 and 1 corresponding to its evidence that “violent explosions” could be an instance of “bombing”.

Semantic Path Checking

One final hypothesis checker we implemented, but did not use, is based on relation expansion through semantic paths. This simple idea is based on the observation that the levels of granularity of the extraction graph and questions often differ, and that many relations admit semantics-preserving “transfers-through” transformations [Lenat and Guha, 1990;

¹ We experimented with more complete geographic resources such as geonames.org, which we found to be overproductive. WordNet’s incomplete, common-sensical coverage seemed to hit the right sweet spot of granularity for our application.

Yeh *et al.*, 2004]. Examples of transfers-through rules include transitivity rules and part ascension rules.

Although the paths are already strongly constrained by the domain and range constraints of relations, the knowledge extracted from the document and the depth limit of search, we have not yet done adequate testing to risk turning on such a permissive checker in the formal evaluation.

4.3 Accepting Support for Hypotheses

Each hypothesis checker provides its measure of confidence that a hypothesis holds, making it possible to learn thresholds for accepting or rejecting hypothesis support. Since we have only very recently received gold standard answers for our question sets, we have not yet set up the training to find good confidence thresholds. For the current evaluation, we universally accept support from the extraction graph and knowledge base inferences over extractions. We set Watson and TyCor confidence thresholds at 0.2 and 0.5 by inspecting results of the system on the dry run corpus and question set, without looking at the final evaluation corpus or queries.

5 Query Answering

5.1 Generating Answer Graphs

The query answering machinery expects input in the form of an RDF extraction graph. For hypotheses supported above threshold, we generate RDF statements that are added to the extraction graph for the target corpus, making hypothesized statements identical in form to those extracted by the NLP stack for the target corpus. For supported type hypotheses, we generate RDF type triples and confidence triples. For supported relation hypotheses, we generate relation triples, confidence triples and several provenance triples (for the provenance string, beginning and ending offsets and the document ID within the corpus).

In the case of hypotheses generated from subqueries of length $N-1$, each RDF graph addition will result in a new answer to the original query. For hypotheses generated from subqueries of length $N-2$ or shorter, hypothesis additions *may* combine to produce new query answers, but this is not guaranteed.

5.2 Results

Extending the knowledge in the extraction graph with supported hypotheses may produce answers to original queries that had no solutions, or produce more answers to queries that did. This obviously does not mean an increase in Recall, as the hypotheses, even if supported, might be incorrect. Even correct hypotheses may combine with extraction errors and produce new incorrect answers to queries.

We looked first at whether our hypothesis-based extension produces answers for queries that originally had none. The numbers of queries with answers on the official evaluation appear in Table 2. The different rows correspond to different thresholds on Watson’s confidence. For the final evaluation, we set the confidence threshold at 0.20, based on inspection of answer correctness on the *dry run* questions and corpus (see section 2).

<i>graph additions</i>	<i>queries with answers</i>	<i>extra answers</i>
<i>no hypotheses</i>	15	--
Watson@0.01	35	~3700
Watson@0.05	31	~1900
Watson@0.10	30	~1100
Watson@0.20	30	~650

Table 2. Number of answerable queries and number of extra answers with hypotheses added

Of course, increasing the number of answers produced does more harm than good if the answers are incorrect. To measure correctness, we ran the *final evaluation* questions on a baseline graph (no extensions from hypotheses) and on a hypothesis-extended graph. The results are in Table 3. The first pair of rows (“*Total*”) show the number of answers for both baseline (B) and hypothesis-extended (H) graphs, followed by the number of True Positive (correct) answers, the number of False Positives (incorrect), the number of False Negatives (answers in the gold standard missed by our system), Precision, Recall and F-Measure. The next four pairs of rows divide the question set into different “Use Cases”. These are categories of questions involving different combinations of relations from the ontology, UC3 involves relations between people, UC4 between people, organizations, and GPEs, and UC5&6 involve events, their participants, locations and times. In all cases, the hypothesis-extended graph produced significantly more answers, as designed. The increase in precision overall (and in UC3&4) shows that the increase in correct answers (TP) was more significant than the increase in incorrect answers (FP). The two use cases where precision dropped still saw a significant enough increase in recall to result in an increase in F-measure. In all cases, the hypothesis-extended graph produced a dramatic increase in F-measure. (The least dramatic increase was in UC6, which had the highest baseline F-measure).

6 Related Work

Modifying queries to control precision and/or recall is an old and well-explored idea. The area of *Query Expansion*

		<i>Ans</i>	<i>TP</i>	<i>FP</i>	<i>FN</i>	<i>P</i>	<i>R</i>	<i>F1</i>
<i>Total</i>	<i>B</i>	139	35	104	507	.25	.06	.10
	<i>H</i>	517	182	335	369	.35	.33	.34
UC3	<i>B</i>	100	13	87	312	.13	.04	.06
	<i>H</i>	369	107	262	218	.29	.33	.31
UC4	<i>B</i>	28	11	17	154	.39	.07	.11
	<i>H</i>	125	56	69	109	.45	.34	.39
UC5	<i>B</i>	3	3	0	23	1.0	.12	.21
	<i>H</i>	11	9	2	26	.82	.26	.39
UC6	<i>B</i>	8	8	0	18	1.0	.31	.47
	<i>H</i>	12	10	2	16	.83	.38	.53

Table 3. Answer correctness for baseline system (B) vs. hypothesis-based QA (H). UC3-6 refer to question categories (Use Cases) in the final evaluation scenario.

[Efthimiadis, 1996] investigates adding² terms to queries as either conjuncts (often to improve precision by disambiguating original query terms) or disjuncts (to improve recall by complementing original query terms with related terms). Terms to add may be found through corpus analysis (for example, finding terms that co-occur with query terms), through relevance feedback or through existing semantic resources such as thesauri or ontologies.

Ontology-based *Query Relaxation* [Hurtado *et al.*, 2008] seeks to improve recall not by deleting query terms, but by “relaxing” them through generalization in a taxonomy of relations and types. For example, an original query to find all people *married to actors* could be relaxed via superrelations (all people *related to actors*) or via supertypes (all people married to **entertainers**) or both. In our system, the Knowledge Base hypothesis checker allows more specific relations and types to satisfy hypotheses based on taxonomic rules from the ontology. It could easily reverse those rules to allow evidence of more general relations and types to support hypotheses.

In *Query Approximation* [Stuckenschmidt and van Harmelen, 2002], a complex, conjunctive query is evaluated by starting with a simpler query (typically a single conjunct) and successively adding query terms. At each step, the set of solutions includes the solutions to the original, complex query, along with (successively fewer) false positives. The goal is to produce these approximate solutions quickly, while converging on exact solutions given more time.

7 Conclusions

A traditional view of NLP is as a layer below that of knowledge representation from which information flows strictly upwards. In this view, NLP components must therefore make their best guess, without any knowledge of the specific task at hand. In this paper, we have presented an approach that improves on this traditional view, by allowing the knowledge layer to “go back” to the NLP layer and ask for more targeted information about certain missing information that will help it complete its task.

Acknowledgments

The authors gratefully acknowledge the support of Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0172. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, AFRL, or the US government.

References

- [Barker et al., 2001] Ken Barker, Peter Clark and Bruce Porter. A Library of Generic Concepts for Composing Knowledge Bases. In *Proceedings of the First International Conference on Knowledge Capture*, pages 14–21, Victoria, Canada, October 2001.
- [Efthimiadis, 1996] Efthimis N. Efthimiadis. Query expansion. *Annual Review of Information Systems and Technology*, 31:121–187, 1996.
- [Ferrucci et al., 2010] David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A. Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John Prager, Nico Schlaefer and Chris Welty. Building Watson: An Overview of the DeepQA Project. *AI Magazine*, 31(3):59–79, Fall 2010.
- [Florian et al., 2004] Radu Florian, Hany Hassan, Abraham Ittycheriah, Hongyan Jing, Nanda Kambhatla, Xiaoliang Luo, Nicolas Nicolov and Salim Roukos. A Statistical Model for Multilingual Entity Detection and Tracking. In *Proceedings of HLT-NAACL*, pages 1-8, Boston, May, 2004.
- [Glass et al., 2009] Michael Glass, Ken Barker, Rekha Kumar, Guhan Ravi and Bruce Porter. Constructing a semantic interpreter using distributional analysis. In *Proceedings of the 8th Conference of the Pacific Association for Computational Linguistics*, Sapporo, 2009.
- [Hurtado et al., 2008] Carlos A. Hurtado, Alexandra Poulou-vassilis and Peter T. Wood. Query relaxation in RDF. *Journal on Data Semantics X*, Lecture Notes In Computer Science 4900:31–61, Springer-Verlag, Berlin.
- [Kipper-Schuler, 2005] Karin Kipper-Schuler. VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon. Ph.D. thesis, Computer and Information Science Department, University of Pennsylvania, Philadelphia, PA.
- [Lenat and Guha, 1990] Douglas B. Lenat and R. V. Guha. *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project*. Addison-Wesley, Reading, Massachusetts, 1990.
- [Pustejovsky, 1995] James Pustejovsky. Linguistic Constraints on Type Coercion, in P. Saint-Dizier and E. Viegas (eds.) *Computational Lexical Semantics*, Cambridge University Press. 1995.
- [Stuckenschmidt and van Harmelen, 2002] Heiner Stuckenschmidt and Frank Van Harmelen. Approximating Terminological Queries. In *Proceedings of the 4th International Conference on Flexible Query Answering Systems*, pages 329–343, Copenhagen, Denmark, October 2002, Springer-Verlag.
- [Suchanek et al., 2007] Fabian M. Suchanek, Gjergji Kasneci and Gerhard Weikum. Yago - A Core of Semantic Knowledge. In *Proceedings of the 16th International World Wide Web Conference*, Banff, Canada, May 2007. ACM Press.
- [Yeh et al., 2004] Peter Yeh, Bruce Porter and Ken Barker. Mining Transformation Rules for Semantic Matching. In *Proceedings of the Workshop on Mining Graphs, Trees and Sequences*, pages 83–94, Pisa, Italy, September 2004.

² Query expansion may also involve *deleting* terms.

Information sources for resolving the ambiguities captured in a packed representation

Doo Soon Kim, Ken Barker and Bruce Porter

University of Texas at Austin

Austin, TX

{onue5, kbarker, porter}@cs.utexas.edu

Abstract

This paper presents Machine Reading architecture based on *packed representation*. We also discuss several information sources useful for resolving the ambiguities captured in the packed representation.

1 Introduction

Building a machine reading system requires combining multiple steps such as parsing, word sense disambiguation, semantic relation assignment, and knowledge integration. One approach to combining these steps is the pipeline architecture, in which the steps are serially connected and each one passes only a *single* output to the next one. Because of its simplicity, this architecture has been commonly used.

The pipeline architecture, however, has one significant problem – aggressive pruning. A component is often forced to make a decision even in the absence of sufficient evidence, which could be available downstream in the pipeline or could even be found by reading subsequent texts. Furthermore, once errors are made by components upstream, subsequent components may compound these errors.

Our approach to this problem is to allow the system to maintain multiple candidate interpretations, thus avoiding premature pruning. The system then chooses the overall best interpretation later when sufficient evidence accumulates. This approach should improve the accuracy of semantic interpretation by allowing the system to resolve ambiguities with more evidence.

A key challenge in this approach is managing the combinatorial explosion of candidate interpretations. The number of candidates is often too great to be simply enumerated. To address this challenge, we previously developed a representation scheme, *packed representation*, which can represent a myriad of candidates succinctly. Using the packed representation, the system can maintain multiple candidates efficiently without suffering from the effects of combinatorial explosion.

Based on this architectural framework, this paper discusses several types of information that could be useful for resolving the ambiguities captured in the packed representation, including redundancy across multiple sentences [Kim *et al.*, 2010], a semantically annotated corpus [Kim *et al.*, 2011] and a knowledge base automatically constructed from texts.

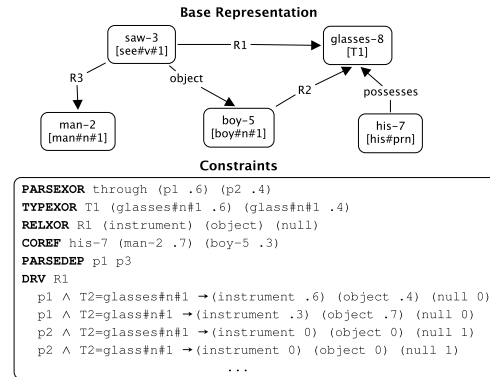


Figure 2: Packed representation

This paper is organized as follows. We present our proposed architecture which is based on the packed representation, and then discuss information sources useful for disambiguating the packed representation (Section 2). Then, we present our current work on automatically building a lexical knowledge base (Section 3).

2 Architecture

Fig. 1 (on the next page) shows the architecture of our proposed system. The system reads a corpus of texts, extracts the contents, and then builds an inference-capable knowledge base.

The system uses the pipeline architecture for assembling the components but, unlike the traditional approach, which maintains only a single interpretation, our architecture maintains multiple candidates using the packed representation. Then, the reasoner infers the best overall semantic representation at the last step among many candidates. The reasoner also exploits several evidence sources that may provide useful information for choosing correct interpretations from the packed representation.

2.1 Packed Representation

In this section, we explain our packed representation introduced in [Kim *et al.*, 2011]. Fig. 2 shows an example of a packed representation for the interpretation of sentence *S1*: “The man saw the boy with his glasses”. The packed representation consists of two parts, the base representation and

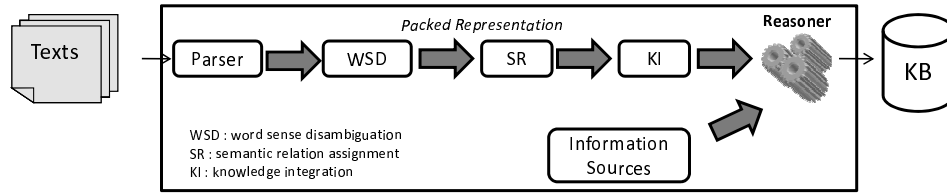


Figure 1: The architecture of our system: it maintains multiple candidate interpretations using the packed representation.

the constraints. The base representation is a graphical representation with two types of semantic information – word senses (in the nodes) and semantic relations between pairs of words (in the edges). A variable is introduced to represent an ambiguous interpretation. For example, the variable T1 represents the fact that the system has not yet committed to a specific word sense for *glasses*.

Constraints are of two types: ambiguity constraints and relationship constraints. Ambiguity constraints enumerate the possible candidates for each variable; relationship constraints describe the relationship among the candidates. We first explain the ambiguity constraints.

Parse Ambiguity

This constraint describes parsing ambiguities. For example, (PARSEXOR with (p1 .6) (p2 .4)) describes the ambiguity of attaching the *with* prepositional phrase. p1 denotes the attachment to *saw* (with score .6) and p2 denotes the attachment to *boy* (with score .4)¹.

Word Sense Ambiguity

This constraint describes ambiguities when assigning a word sense. For example, (TYPEXOR T1 (glasses#n#1 .6) (glass#n#1 .4)) represents that the word sense for *glasses* can be glasses#n#1 (with the score .6) or glass#n#1 (with the score .4), but not both.

Semantic Relation Ambiguity

This constraint describes ambiguities in assigning a semantic relation. For example, (RELXOR R1 (instrument) (object) (null)) represents the fact that there are three candidates for the semantic relation between *saw* and *glasses*: instrument, object, and null (no relation). This example does not include numerical scores but instead uses a DRV constraint (see below) to describe how the scores can be calculated from the other constraints.

Co-reference Ambiguity

This constraint describes ambiguities when assigning a co-reference link. For example, (COREF his-7 (man-2 .7) (boy-5 .3)) represents that *his* may refer to *man* (with score .7) or *boy* (with score .3).

In addition to representing ambiguities, a packed representation can also represent the relationships among them.

¹p₁ and p₂ indicate a set of dependency triples, {(saw-3 prep_with glasses-8)} and {(boy-5 prep_with glasses-8)}, respectively.

²<lemma>#<part-of-speech>#<sense number>

Dependency in Parsing

This constraint describes the dependency between two parse fragments. For example, (PARSEDEP p1 p3) represents that the triples denoted by p1 depend on the ones denoted by p3. Therefore, if any dependency triple in p3 turns out to be false, the triples in p1 should be discarded, too.

Derivation

This constraint describes how the candidates and their scores are derived from the interpretations made upstream. For example, the DRV constraint in fig. 2 says that the candidates and their scores for R1 are derived based on the decision between p1 and p2 (the first PARSEXOR constraint) and the assignment in T1. Each row describes the scores of the candidate relations when the upstream components make a different choice. For example, the first row shows that the scores of instrument, agent and null in R1 are .6, .4 and 0 respectively when p1 is chosen and T1 is glasses#n#1.

Similar to belief maintenance systems, these constraints enable the system to adjust each candidate’s confidence score in response to changes to other variables.

2.2 Reasoner

Given a packed representation, the reasoner infers the most probable interpretations based on the evidence sources. One inference engine that we have used is Alchemy³. In Appendix A, we show how we convert a packed representation into Alchemy statements.

2.3 Evidence Sources

Several types of information could be useful for resolving the ambiguities in the packed representation.

Annotated Corpus

An annotated corpus provides useful statistics about candidate interpretations and the relationship among multiple candidates. We previously evaluated one annotated corpus, OntoNotes ([Hovy *et al.*, 2006]) [Kim *et al.*, 2011]. In our work, we counted the frequency of candidate interpretations (candidate word senses and semantic relations) in OntoNotes for a similar context and then used the count to adjust the ranking of the candidates in the packed representation. To evaluate our approach, we compared two systems, our system, which applied OntoNotes to the packed representation, and the pipeline system, which used OntoNotes but selected only a single output at each step. See [Kim *et al.*, 2011] for the positive evaluation result that shows OntoNotes is useful for disambiguating the packed representation.

³<http://alchemy.cs.washington.edu>

Hand-built Knowledge Bases

Hand-built knowledge bases such as Cyc and Linked Data could be useful. The advantage of using these resources is that their representational quality is good. [Yeh *et al.*, 2006] used a hand-built knowledge base to help interpret sentences. Similar to our approach, their system maintained multiple candidate interpretations and then selected the one most supported by the knowledge base. Unlike our approach, they used a beam rather than a packed representation. Their experimental result was encouraging, showing that a hand-built knowledge base could be useful for ambiguity resolution.

Automatically-built Knowledge Bases

Constructing hand-built knowledge bases requires expertise and is time-consuming. To address this problem, several automated approaches attempt to extract information, such as scripts [Chambers and Jurafsky, 2008], causal/temporal information [Chklovski and Pantel, 2004], and syntactic relationship among the words [Fan *et al.*, 2010]. Unlike hand-built knowledge bases, however, the quality is not as high and they have not been thoroughly evaluated in the context of machine reading.

We evaluated one knowledge resource in this category called Prismatic [Fan *et al.*, 2010], a large-scale resource that stores parse fragments (e.g., (eat prep_with fork)) along with their frequency in given corpora. For each candidate interpretation in the packed representation, we assigned its confidence score using the frequency data from Prismatic. Then, we inferred the most probable semantic representation based on those confidence scores. [Kim, 2011 Forthcoming] explains the method and our evaluation in detail. The evaluation result shows that our initial method to use Prismatic is partially effective – with Prismatic, accuracy of word sense disambiguation drops, while accuracy of semantic relation assignment increases.

Redundant Sentences

We previously showed that redundancy across multiple sentences was an useful information source [Kim *et al.*, 2010]. Redundant sentences express the same meaning and therefore their meaning representation should be similar. Based on this observation, our approach selects the semantic representations that occur most frequently across multiple packed representations generated from redundant sentences.

[Kim *et al.*, 2010] shows that redundancy is useful information for resolving ambiguities in packed representation..

Neighboring Sentences

Consecutive sentences are typically coherently organized to deliver information efficiently. Coherence, therefore, could be a useful information source for interpreting consecutive sentences. For example, a co-reference chain might help to disambiguate the sense of words that are connected through the chain, because they should all share the same sense.

Ultimately, we envision a system that can adjust the ranking of the candidate interpretations while reading through the subsequent sentences and exploiting inter-sentential coherence. A key component in this system would be one that can measure the degree of coherence among multiple meaning representations.

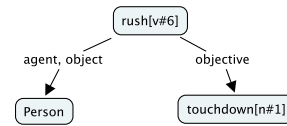


Figure 3: Knowledge representation about rushing: a person performs rushing (wordnet sense : v#6) to score a touchdown.

3 Building a Lexical Knowledge Base from Texts

We are currently developing an algorithm for building a lexical knowledge base from texts. This knowledge base will be used as another evidence source by our system. Fig. 3 shows an example of a knowledge structure that the algorithm produces. The lexical knowledge is semantically represented using the word senses and the semantic relations.

We present the algorithm step-by-step using an example of learning a lexical item, rush (in a football game).

Step 1. Gather sentences containing the target lexeme

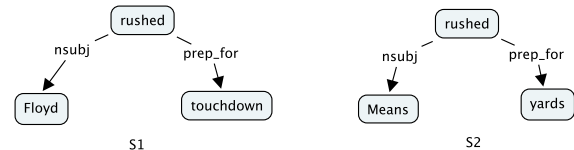
The example sentences containing “rush” are :

S1. William Floyd rushed for three touchdowns, moving the 49ers one victory from the Super Bowl.

S2. San Diego’s Natrone Means rushed 24 times for 139 yards, including a 24-yard touchdown run in the third quarter.

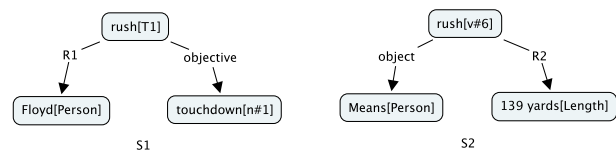
Step 2. Extract the parse fragments around the lexeme

This step extracts the context around the lexeme in the parse trees of the sentences. In our example, we define the context as the dependency triples whose dependency label is one of subject, object, or preposition and contains the target lexeme.



Step 3. Produce the packed representation

The parse fragments produced by step 2 are converted into packed representations. The named instances in the representations are replaced by their semantic types.



* The constraints are omitted for both representations

3. Combine the packed representations

These packed representations represent parts of the same semantic knowledge about the target lexeme. Therefore, if same semantic representations appear redundantly across multiple packed representations, we hypothesize that they tend to be correct. This observation is similar to the one used in [Kim *et al.*, 2010] which combines redundant sentences.

Similar to the method in [Kim *et al.*, 2010], this step combines the redundant interpretations across multiple packed

representations to increase their confidence scores. For example, assuming that T1 in S1 has a candidate type, MOVE, and that R1 has a candidate semantic relation, *object*, the confidence scores of MOVE and *object* in (rush[Move] object Person) would be increased because the triple appears in both representations. After all packed representations are combined into a single representation, the overall best representation is extracted. Figure 3 shows the result of this step.

4 Conclusion

We presented an architecture based on packed representation to address the aggressive pruning of the pipeline architecture. We also presented several types of information sources that could be useful for resolving ambiguities in the packed representation. Finally, we presented an algorithm for automatically building a lexical knowledge base from texts.

A Translation to Alchemy

This appendix describes the translation of our packed representation to the Alchemy statements.

PARSEXOR($p_1, p_2, \dots, p_n$)

If ϕ^4 is included in PARSEXOR,

$$p_i \rightarrow!(p_1 \wedge \dots p_{i-1} \wedge p_{i+1} \dots \wedge p_n). \text{ for all } i \in 1, \dots, n$$

Otherwise,

$$p_i \leftrightarrow!(p_1 \wedge \dots p_{i-1} \wedge p_{i+1} \dots \wedge p_n). \text{ for all } i \in 1, \dots, n$$

p_i is a binary variable that takes 1 (if the corresponding dependency triples are true) or 0 (otherwise). If ϕ is included in PARSEXOR, p_i cannot be confirmed even though the others are found to be false. If ϕ is not included, the above statement expresses a mutually exclusive relationship among p_i s: p_i is true if and only if the other variables are false.

$$1 \text{ sent} \rightarrow p_i \text{ if the corresponding dependency triples appear in the top-scored parse}$$

$$.3 \text{ sent} \rightarrow p_j \text{ otherwise}$$

The above statements express the preference for the top-scored candidate parse *a priori*. 1 and .3 are the weights. If there is no evidence to override this preference, the system will choose the dependency triples from the top-scored parse. *sent* is a predicate defined as evidence.

TYPEXOR($w_i, (t_1 s_1), (t_2 s_2), \dots, (t_n s_n)$)

$$ws_w_i(< variable >!) \rightarrow$$

$$\log s_i \quad sent \rightarrow ws_w_i(t_i) \text{ for all } i \in 1, \dots, n$$

$ws_w_i(t_j)$ represents that t_j is the sense of the word w_i . The first rule (defined in the header) asserts that there should be only one sense for w_i . The second rule specifies the weight ($\log s_i$) for each candidate sense.

⁴ ϕ indicates the empty set. PARSEXOR($p_1, p_2, \dots, \phi$) means that if p_i is correct, the others are wrong, but does not mean that if all but p_i are incorrect, p_i is correct. See [Kim et al., 2011].

RELXOR($w_i w_j (r_1 s_1), (r_2 s_2), \dots, (r_n s_n)$)

$$rel_w_i_w_j(< variable >!) \rightarrow$$

$$\log s_i \quad sent \rightarrow rel_w_i_w_j(r_i) \text{ for all } i \in 1, \dots, n$$

$rel_w_i_w_j(r_k)$ represents that r_k is a semantic relation connecting w_i and w_j .

PARSEDEP(p_1, p_2)

$$p_1 \rightarrow p_2.$$

This rule states that if p_2 is false, p_1 should be false.

DRV($R_1, (p_1, t_{11}, t_{12}, (r_{11} s_{11}), \dots, (r_{1m} s_{1m})), \dots, (p_n, t_{n1}, t_{n2}, (r_{n1} s_{n1}), \dots, (r_{nm} s_{nm}))$)

$$\log s_{ij} \quad p_i \wedge ws_w_1(t_{i1}) \wedge ws_w_2(t_{i2}) \rightarrow rel_w_1_w_2(r_{ij})$$

$$\text{for all } i \in \{1, \dots, n\} \setminus \{NIL\}$$

$$p_a \vee \dots \vee p_z \leftrightarrow !rel_w_1_w_2(NIL).$$

The first rule directly represents the derivation relationship except for the NIL candidate. It represents that if p_i , t_{i1} and t_{i2} are true, r_{ij} is a correct semantic relation with the weight $\log s_{ij}$. In the second rule, $p_a, \dots, p_z$ denote a parse fragment containing a dependency triple connecting w_1 and w_2 . The rule says that any of $p_a, \dots, p_z$ is true if and only if a semantic relation exists between w_1 and w_2 .

References

- [Chambers and Jurafsky, 2008] Nate Chambers and Dan Jurafsky. Unsupervised learning of narrative event chains. In *ACL*, 2008.
- [Chklovski and Pantel, 2004] Timothy Chklovski and Patrick Pantel. Verbocean: Mining the web for fine-grained semantic verb relations. *ACL*, 2004.
- [Fan et al., 2010] James Fan, David Ferrucci, David Gondek, and Aditya Kalyanpur. Prismatic: inducing knowledge from a large scale lexicalized relation resource. In *Proc. of NAACL HLT 2010 Workshop*, 2010.
- [Hovy et al., 2006] Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. OntoNotes: The 90% Solution. In *HLT/NAACL*, 2006.
- [Kim et al., 2010] Doo Soon Kim, Ken Barker, and Bruce Porter. Improving the quality of text understanding by delaying ambiguity resolution. In *CoLing*, 2010.
- [Kim et al., 2011] Doo Soon Kim, Ken Barker, and Bruce Porter. Delaying ambiguity resolution in the pipeline architecture to avoid aggressive pruning. Technical Report TR-11-24, University of Texas at Austin, 2011.
- [Kim, 2011 Forthcoming] Doo Soon Kim. *Knowledge integration for machine reading*. PhD thesis, University of Texas at Austin, 2011 (Forthcoming).
- [Yeh et al., 2006] Peter Yeh, Bruce Porter, and Ken Barker. A unified knowledge based approach for sense disambiguation and semantic role labeling. In *AAAI*, 2006.

Learning Rules from Incomplete Examples via Observation Models

Janardhan Rao Doppa, Mohammad NasrEsfahani, Mohammad S. Sorower, Jed Irvine
Thomas G. Dietterich, Xiaoli Fern, and Prasad Tadepalli

School of EECS, Oregon State University
Corvallis, OR 97330, USA

{doppa,nasresfm,sorower,irvine,tgd,xfern,tadepalli}@cs.orst.edu

Abstract

We study the problem of learning general rules from concrete facts extracted from natural data sources such as the newspaper stories and medical histories. Natural data sources present two challenges to automated learning, namely, *radical incompleteness* and *systematic bias*. In previous work we proposed an approach that combines simultaneous learning of multiple predictive rules with differential scoring of evidence based on implicit observation models to address the above problems. In this paper, we further evaluate our approach empirically on natural datasets based on both textual and non-textual sources. We present a theoretical analysis that elucidates our approach and explains the empirical results.¹

1 Introduction

Learning common sense knowledge in the form of rules by reading from natural texts has long been a dream of AI [Guha and Lenat, 1990]. This problem presents an opportunity to exploit the long strides of research progress made in natural language processing and machine learning in recent years [Nahm and Mooney, 2000; Carlson *et al.*, 2010; Schoenmackers *et al.*, 2010].

Unfortunately there are two major obstacles to fully realizing the dream of robust learning of general rules from natural sources. First, natural data sources such as texts and medical histories are *radically incomplete* — only a tiny fraction of all true facts are ever mentioned. More importantly, natural sources are *systematically biased* in what is mentioned. In particular, news stories emphasize newsworthiness, which correlates with rarity or novelty, sometimes referred as “the man bites dog phenomenon.” For example, consider the following sentence in a real news story:

“Ahmed Said Khadr, an Egyptian-born Canadian, was killed last October in Pakistan.”

Presumably, the phrase “Egyptian-born” was considered important by the reporter because it violates the expectation

that most Canadians are born in Canada. The birth place would most likely have been omitted if it was Canada.

In previous work, learning from incomplete examples or partial assignments has been studied under noise-free settings in the probably approximately correct learning framework. The goal is to learn an approximation of a function that has a small error with respect to the training distribution from incompletely described examples. It has been shown that the sample complexity of finite hypothesis spaces remains the same under incomplete examples as under complete examples [Khaddon and Roth, 1999]. Further, when the hypothesis space obeys certain conditions such as “shallow monotonicity,” the problem of learning from incomplete examples polynomially reduces to that of learning from complete examples [Michael, 2009]. In fact, the same learning algorithm can be used after the missing data is completed in a way that guarantees consistency with the target function. This approach is validated on an extensive study of sentence completion tasks on a natural dataset [Michael and Valiant, 2008].

Our approach to learning from incomplete examples extends the above work in multiple directions. First, we learn multiple rules for several predicates simultaneously and use the rules to complete the missing data to improve learning. We call this approach “multiple predicate bootstrapping (MPB)” [Doppa *et al.*, 2010]. Second, we adopt the above approaches to noisy observations and systematic bias.

Our main solution to deal with systematic bias is to differentially score the evidence for rules based on a presumed model of observation. In the “missing at random” (MAR) model [Little and Rubin, 1987; Jaeger, 2006], data is omitted based only on information that is already mentioned. In this case, conservative scoring of evidence, where rules are only evaluated when all relevant data is present, gives an unbiased estimate of the rule correctness. In the *novelty mention model*, a special case of “missing not at random” (MNAR) model and illustrated by the above Egyptian-born Canadian example, data is mentioned with a higher probability if it cannot be inferred from the remaining data. We show that under this model, aggressive scoring of rules, where we count evidence against a rule only if it contradicts the rule regardless of how the missing information transpires, gives a better approximation to the accuracy of the rule. Our empirical results compare favorably to baselines such as EM and Structural EM and are consistent with the theoretical predictions.

¹This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL) under Contract No. FA8750-09-C-0179.

2 Multiple-Predicate Bootstrapping

Algorithm 1 Multiple-Predicate Bootstrapping (MPB)

Input: $\mathcal{D}_I$ = Incomplete training examples, $\mathcal{M}$ = Implicit mention model, τ = support threshold, θ = confidence threshold

Output: set of learned rules $\mathcal{R}$

```

1: repeat
2:   LEARN RULES:  $\mathcal{R} = \emptyset$ 
3:   for each hypothesized rule  $r$  do
4:     compute support  $\tau_r$  and confidence  $\theta_r$  of the rule  $r$ 
       using  $\mathcal{D}_I$  and implicit mention model  $\mathcal{M}$ 
5:     if  $\tau_r > \tau$  and  $\theta_r > \theta$  then  $\mathcal{R} = \mathcal{R} \cup \{r\}$ 
6:   end for
7:   IMPUTE MISSING DATA:
8:   for each missing fact  $f_m \in \mathcal{D}_I$  do
9:     Predict  $f_m$  using the most-confident applicable rule
        $r \in \mathcal{R}$ 
10:    if  $f_m$  is predicted then  $\mathcal{D}_I = \mathcal{D}_I - \{f_m\}$ 
11:  end for
12: until convergence
13: return the set of learned rules  $\mathcal{R}$ 

```

Our algorithmic approach, called “Multiple-Predicate Bootstrapping,” (MPB) is inspired by several lines of work including co-training [Blum and Mitchell, 1998], multitask learning [Caruana, 1997], coupled semi-supervised learning [Carlson *et al.*, 2010], and self-training [Yarowsky, 1995]. It simultaneously learns a set of rules for each predicate in the domain given other predicates and then applies the learned rules to impute missing facts in the data. This is repeated until no new fact can be added. Following the data mining literature, we evaluate each rule using two measures: support and confidence. The support of a rule is measured by the number of examples that satisfy the body of the rule. The higher the support, the more statistical evidence we have on the predictive accuracy of the rule. In order to use a rule to impute facts, we require its support to be greater than a *support threshold*. We measure the confidence of a rule as the ratio of the number of records that satisfy both body and head of the rule to the number that satisfy the body, which estimates the conditional probability of the head of the rule given its body.

We use a relational data mining algorithm called FARMER [Nijssen and Kok, 2003] for learning rules. FARMER systematically searches the space of possible rules up to a fixed depth d (candidate rules) whose support and confidence exceed the given thresholds using depth first search. Its advantages over other rule learning systems such as FOIL are that a) it can learn redundant rules, which is useful with incomplete data; and b) it has the flexibility to vary the depth for search efficiency. Given multiple learned rules that are applicable to a given instance, we use the “most confident” one to make predictions. The overall algorithm is summarized in Algorithm 1.

Implicit Mention Models. We address the problem of systematic bias by adapting the scoring function for the hypothesized rules according to a presumed *implicit* mention model.

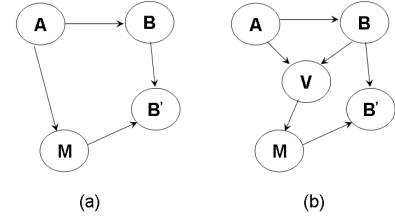


Figure 1: Bayes nets for data generation using (a) Random Mention Model (b) Novelty Mention Model. $A \Rightarrow B$ is a rule, M is a random variable that represents the fact that B is mentioned, B' indicates the observed value of B and random variable V denotes the violation of rule.

We now discuss two specific mention models and two methods for scoring evidence for rules.

Random Mention Model (RMM): This is equivalent to the Missing At Random (MAR) statistical model. In this model, it is assumed that facts are mentioned based on other known facts but not based on missing facts. For example, a doctor might omit a test if some other tests come out negative. A Bayesian network that illustrates this case is shown in Figure 1(a). B' is equal to B if M is true.

Novelty Mention Model (NMM): In this model, facts that are not entailed by the previously mentioned facts and prior knowledge are more likely to be mentioned. This is a special case of Missing Not At Random (MNAR) statistical model, since whether a fact is missing depends on the value of the fact itself as illustrated in Figure 1(b). In the novelty mention model, consider a α -general rule (i.e., a rule with confidence $\geq \alpha$) $A \Rightarrow B$, B will be mentioned with higher probability when the rule is violated, i.e., $P(M|V) > P(M|\neg V)$. Note that for rules that are not α -general, the facts entailed by these rules will not be missing because they are not considered α -general predictable due to the lack of generality of the rules. This model more closely captures the citizenship-birth place example, since whether or not the birth place of a person is mentioned depends on the birth place and other mentioned facts of the person such as the citizenship.

Inspired by the two types of mention models, we propose two different ways of scoring rules. We use the following notation to define our rule scoring functions. Each literal may be either true, false or unknown. We write $n(A = t, B = f, C = u)$ to be the count of examples where A is true, B is false and C is unknown. For brevity we write A for $A = t$. The **Support** of a rule $A \Rightarrow B$ is defined as the number of examples in which A is known to be true i.e., $n(A)$, for both conservative and aggressive scoring.

In *conservative scoring*, evidence is counted in favor of a rule only when all facts relevant to determining the truth value of the rule are actually known. The confidence of the rule in this case is defined as follows:

$$p_c(A \Rightarrow B) = \frac{n(A, B)}{n(A, B \neq u)} \quad (1)$$

In *aggressive scoring*, a fact is counted as evidence for a

rule if the rule's premise is satisfied and the conclusion is not contradicted. The confidence of a rule is defined as follows:

$$p_a(A \Rightarrow B) = \frac{n(A, B) + n(A, B = u)}{n(A)} \quad (2)$$

For example, consider the text "Khadr, a Canadian citizen, was killed in Pakistan". For conservative scoring, it is counted as neither supporting nor contradicting the rule `citizen(X, Y)  $\Rightarrow$  bornIn(X, Y), as we are not told that bornIn(Khadr, Canada). In contrast, it is counted as supporting the rule citizen(Y)  $\Rightarrow$  bornIn(Y) for aggressive scoring because, adding bornIn(Canada) supports the rule without contradicting the available evidence.`

3 Analysis of Implicit Mention Models

This section analyzes aggressive and conservative scoring of data generated using different mention models.

Consider a rule $A \Rightarrow B$. Figure 1 shows the Bayes nets that explains the data generation process in the random and novelty mention models. Let S be the support set of the rule $A \Rightarrow B$, i.e., the set of examples where A is true. Let $p(r)$ be the true confidence of the rule r , i.e., the conditional probability of B given A . Let $\hat{p}_c(r)$ and $\hat{p}_a(r)$ denote the conservative and aggressive estimates of confidence of the rule r .

Theorem 1. *If the data is generated by the random mention model then $\hat{p}_c(r)$ is an unbiased estimate and $\hat{p}_a(r)$ is an overestimate of the true confidence of rule $p(r)$.*

Proof. Conservative scoring estimates the confidence of the rule from only a subset of S where B is not missing.

$$\hat{p}_c(r) = \frac{|S| p(r) P(M|A)}{|S| P(M|A)} = p(r) \quad (3)$$

Therefore, $\hat{p}_c$ is a unbiased estimate of the true confidence.

Aggressive scoring deterministically imputes the missing value of B such that it satisfies the hypothesized rule.

$$\begin{aligned} \hat{p}_a(r) &= \frac{|S| p(r) P(M|A) + |S| (1 - P(M|A))}{|S|} \\ &= p(r) P(M|A) + (1 - P(M|A)) \\ &= p(r) + (1 - P(M|A)) (1 - p(r)) \\ &\geq p(r) \end{aligned} \quad (4)$$

Therefore, $\hat{p}_a(r)$ overestimates the confidence of the rule. The bias of $\hat{p}_a(r)$ increases with decreased $P(M|A)$. $\square$

Theorem 2. *If the data is generated by the random mention model, then the true ranking order of rules is preserved with both conservative and aggressive scoring.*

Proof. It is enough to show that the ordering is preserved for any two rules r_1 and r_2 that predict the value of the same variable. Without loss of generality, let $p(r_1) > p(r_2)$. From (3), $\hat{p}_c(r_1) > \hat{p}_c(r_2)$. Therefore, order is preserved with conservative scoring.

$$\begin{aligned} p(r_1) &> p(r_2) \\ \Rightarrow p(r_1) P(M|A) + (1 - P(M|A)) &> p(r_2) P(M|A) + (1 - P(M|A)) \\ &> p(r_2) P(M|A) + (1 - P(M|A)) \\ \Rightarrow \hat{p}_a(r_1) &> \hat{p}_a(r_2) \quad (\text{From (4)}) \end{aligned}$$

Thus, aggressive scoring also preserves the ordering. $\square$

Theorem 3. *If the data is generated by the novelty mention model, then $\hat{p}_c(r)$ is an underestimate and $\hat{p}_a(r)$ is an overestimate of true confidence of the rule $p(r)$.*

Proof. In what follows V stands for a random variable that represents a violation of a confident rule in predicting B . If V is true, according to the novelty model, B has a higher probability of being mentioned. Hence $P(M|V) > P(M|\neg V)$.

$$\begin{aligned} \hat{p}_c(r) &= \frac{|S| p(r) P(M|\neg V)}{|S| p(r) P(M|\neg V) + |S| (1 - p(r)) P(M|V)} \\ &= \frac{p(r) P(M|\neg V)}{p(r) P(M|\neg V) + (1 - p(r)) P(M|V)} \end{aligned}$$

To compare this with $p(r)$, we estimate the odds:

$$\begin{aligned} &\frac{\hat{p}_c(r)}{1 - \hat{p}_c(r)} \\ &= \frac{p(r) P(M|\neg V)}{(1 - p(r)) P(M|V)} \\ &= \text{true odds} \times \frac{P(M|\neg V)}{P(M|V)} \\ &< \text{true odds} \end{aligned}$$

Since in the novelty mention model $P(M|\neg V) < P(M|V)$, $\hat{p}_c(r)$ underestimates $p(r)$ and significantly so if $P(M|\neg V) \ll P(M|V)$.

It is easy to show that for aggressive scoring, we have:

$$\begin{aligned} \hat{p}_a(r) &= \frac{|S| p(r) + |S| (1 - p(r)) (1 - P(M|V))}{|S|} \\ &= p(r) + (1 - p(r)) (1 - P(M|V)) \\ &\geq p(r) \end{aligned} \quad (5)$$

$\square$

Therefore, similar to the random mention model, $\hat{p}_a(r)$ overestimates the true confidence of the rule $p(r)$. However, when the novelty mention model is strongly at play, i.e., $P(M|V) \approx 1$, it provides a good estimate of $p(r)$.

Theorem 4. *If the data is generated by the novelty mention model, then the true ranking order of the rules is preserved with aggressive scoring.*

Proof. We first show that the ordering is preserved for any α -general rules r_1 and r_2 where $p(r_1) > p(r_2)$.

$$\begin{aligned} p(r_1) &> p(r_2) \\ \Rightarrow p(r_1) P(M|V) &> p(r_2) P(M|V) \\ \Rightarrow p(r_1) P(M|V) + (1 - P(M|V)) &> p(r_2) P(M|V) + (1 - P(M|V)) \\ \Rightarrow \hat{p}_a(r_1) &> \hat{p}_a(r_2) \quad (\text{From (5)}) \end{aligned}$$

We then compare an α -general rule r_1 with a rule r_2 that is not α -general. For r_1 , $\hat{p}_a(r_1) \geq p(r_1)$ over-estimates the confidence based on Theorem 3. For r_2 , because no data is missing, $\hat{p}_a(r_2) = p(r_2)$ is an unbiased estimate of $p(r_2)$. $\hat{p}_a(r_1) \geq p(r_1) > p(r_2) = \hat{p}_a(r_2)$. Thus, r_1 will be correctly ranked higher than r_2 by aggressive scoring. Finally

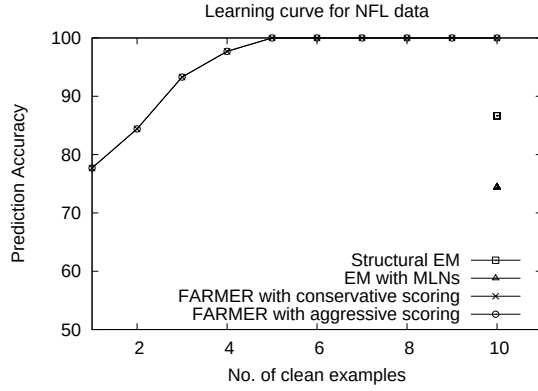


Figure 4: Results of NFL domain: no. of clean examples vs. prediction accuracy

consider two rules that are both not α -general, because there is no missing data, aggressive scoring provides unbiased estimate of the confidences and preserves their rank order. $\square$

It is interesting to note that while conservative scoring preserves the ranking order of α -general rules, it can potentially reverse the order of an α -general rule with a rule that is not α -general. This is because conservative scoring correctly estimates the confidence of rules that are not α -general but underestimates the confidence of the α -general rules.

4 Experimental Results

In this section, we describe our experimental results with both synthetic and natural datasets and carefully analyze them.

Synthetic Experiments. To test our analysis of implicit mention models, we perform experiments on synthetic data generated using different missing mechanisms, i.e., RMM and NMM. We use the UCI database *SPECT Heart*, which describes diagnosing of Single Proton Emission Computed Tomography (SPECT) images². This database contains 267 examples with 23 binary features extracted from the SPECT image sets (patients). A 70% / 30% split of the data is created for training and testing respectively. We generate two different synthetic versions based on RMM and NMM missing mechanisms (see Figure 1). We first learn a set of deterministic rules from the training data, and then retain those that have a confidence of 80% or more. These rules are then used to create training and testing data with varying levels of missingness. For NMM, if some rule is violated, then the consequent is always mentioned. If no rule is violated, then the consequent is omitted based on the missingness level. We evaluate the learning algorithms on the test data generated by the same mention model that generates its training data. Experiments are performed with different levels of missingness in both training and testing data and we report the accuracies (averaged over all attributes) with which the missing data is predicted correctly w.r.t the gold standard data. The averaged

results over 10 different versions of the generated data are reported (see Table 1 and Table 2).

Baselines: We compare the results of our Multiple-Predicate Bootstrapping (MPB) approach using the relational data mining algorithm FARMER [Nijssen and Kok, 2003] with Structural EM (SEM) [Friedman, 1998] and EM implemented using Markov Logic Networks (MLNs). Structural EM learns both structure and parameters of the Bayes net from incomplete data, and then the learned Bayes net is used to make predictions on the test dataset. To run EM, we initialize an MLN with high support rules learned from incomplete data and learn its weights generatively [Richardson and Domingos, 2006].

Analysis of Results: For the RMM data, both conservative and aggressive scoring perform equally well (see Figure 2(a)), which was expected based on Theorem 2. Since the ranking order of rules is the same in both scoring methods, they make the same prediction by picking the same rule that is applicable. SEM performs better than both conservative and aggressive when missingness in training data is small, i.e., 0.2 and 0.4 (see $\square$'s in Figure 2(b)), but conservative/aggressive scoring significantly outperforms SEM when missingness in training data is large, i.e., 0.6 and 0.8 (see $\diamond$'s in Figure 2(b)). Performance of all the algorithms decreases as the percentage of missingness in training data increases.

For the NMM data, aggressive scoring significantly outperforms conservative scoring (see Figure 3(a)) which is consistent with our analysis in Theorem 4. Since the novelty mention model was strongly at play, i.e., $P(M|V) \approx 1$, aggressive scoring provides a very good estimate of the true confidence of the rules, resulting in excellent performance. Aggressive scoring significantly outperforms SEM when the missingness in data is tolerable, i.e., 0.2, 0.4 and 0.6. However, all algorithms including SEM perform poorly with exceedingly high missingness, i.e., 0.8. Note that, although our analysis of implicit mention models is for the simple case where only the head of the rule can be missing, our synthetic data were generated for a more difficult problem where the body of the rule could be missing as well.

Experiments with Real data. We also performed experiments on two real world domains: 1. NFL data, 2. Birthplace-Citizenship data. We used data extracted by BBN's information extraction system on a LDC (Linguistic Data Consortium) training corpus of 110 NFL sports articles and 248 news stories related to the topics of people, organizations and relationships respectively.

For the NFL domain, the following predicates were provided for each game with natural interpretations: *gameWinner*, *gameLoser*, *homeTeam*, *awayTeam*, *gameTeamScore*, and *teamInGame*. A test set of 100 examples was used, to evaluate the performance of the learned rules based on the accuracy of predicting the missing facts w.r.t the ground truth. We observed that most of the input extractions are noisy and inconsistent, that makes the problem of rule learning even harder. These inconsistent examples are due to co-reference errors, e.g., the extractor may not realize that two mentions of the same team in a football article are in fact the same. We want to use this insight while scoring each rule. To address this we learned integrity constraints from a

²<http://archive.ics.uci.edu/ml/datasets/SPECT+Heart>

		Testing											
		0.2			0.4			0.6			0.8		
	Missing %	CONS	AGGR	SEM	CONS	AGGR	SEM	CONS	AGGR	SEM	CONS	AGGR	SEM
Training	0.2	77.8	77.8	81.8	77.9	77.9	81.9	77.8	77.8	81.4	77.5	77.6	80.0
	0.4	76.7	76.7	79.5	77.1	77.1	79.4	77.0	76.9	79.3	76.9	76.8	78.2
	0.6	77.6	77.7	72.2	77.9	78.0	73.3	77.4	77.5	72.9	77.2	77.5	72.5
	0.8	75.4	75.2	70.2	75.6	75.1	71.6	75.0	74.5	71.2	74.9	74.5	70.9

Table 1: Accuracy results of synthetic experiments with Random Mention Model (RMM) data

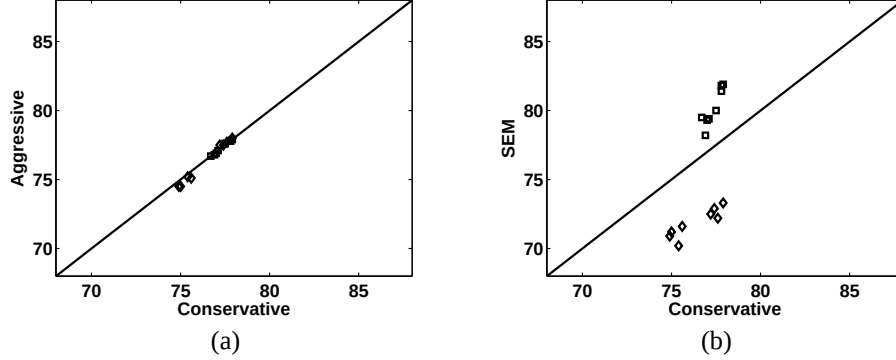


Figure 2: Accuracy for Random Mention Model (RMM) data : (a) Conservative vs. Aggressive (b) Conservative vs. SEM

small number of complete examples, and applied the learned integrity constraints to generate consistent versions of the inconsistent examples (e.g., by deleting a literal) in all possible ways. Finally, we scored the rules against these ‘corrected’ examples with a lower weight $\gamma (< 1)$. The prediction accuracy of the learned rules is reported as a function of number of clean examples (see Figure 4). The results of this approach are compared with Structural EM (SEM) and EM algorithm using Markov Logic Networks (MLNs) initialized with the highest support rules learned from incomplete data. We then perform generative weight learning of MLNs via EM. We use Lazy MCSAT to do MAP inference for predicting the missing facts in the test set. For both SEM and EM, we use the ground-truth (instead of the learned) integrity constraints to correct the noisy examples and hence, there is only one point for each of them in Figure 4.

Analysis of Results: As we can see in Figure 4, both conservative and aggressive scoring significantly outperform both SEM and EM. Since the NFL domain is deterministic, i.e., $\forall r, p(r) = 1$, and similar to RMM data, both conservative and aggressive scoring perform equally. We observed that once we learn the true integrity constraints from the clean examples, conservative scoring exactly learns the ground truth rules and aggressive scoring learns a few other spurious rules as well. However, the ground-truth rules are ranked higher than the spurious rules based on the estimated confidences and therefore, the spurious rules do not degrade the performance. Similar to the results on the synthetic data, SEM does not perform very well when the data is radically incomplete.

Birthplace-Citizenship data: Manual analysis of this training corpus revealed that the birth place of some person is

only mentioned 23 times in the 248 documents. Moreover, in 14 of the 23 mentions, the information violates the default rule $\text{citizen}(Y) \Rightarrow \text{bornIn}(Y)$. Since the data matches the assumption of aggressive scoring, it is expected to learn the correct rule. However, our extracted data was highly noisy and inaccurate. More specifically, the extracted data had 479 examples which mentioned only the citizenship of a person, 4 examples where both birth place and citizenship were mentioned out of which only 1 example violated the default rule. Therefore, confidence of the rule $\text{citizen}(Y) \Rightarrow \text{bornIn}(Y)$ was 0.75 based on conservative scoring and 0.9967 based on aggressive scoring. Since we used a confidence threshold of 0.8 for all our experiments, only aggressive scoring learned the correct rule. We also did this experiment with EM using MLNs and found that its performance was similar to conservative scoring.

5 Conclusions and Future Work

We motivated and studied the problem of learning from natural data sources which presents the dual challenges of radical incompleteness and systematic bias. Our solutions to these problems consist of bootstrapping from learning of multiple relations and scoring the rules or hypotheses differently based on an assumed mention model. Our experimental results validate the usefulness of differential scoring of rules and show that our approach can outperform other state-of-the-art methods such as Structural EM and EM. Our theoretical analysis gives insights into why our approach works, and point to some future directions. One of the open questions is the analysis of multiple-predicate bootstrapping and the conditions under which it works. Another avenue of future research is

		Testing											
		0.2			0.4			0.6			0.8		
	Missing %	CONS	AGGR	SEM	CONS	AGGR	SEM	CONS	AGGR	SEM	CONS	AGGR	SEM
Training	0.2	97.1	98.1	90.0	96.8	97.8	88.0	96.7	97.5	87.0	96.9	97.6	86.0
	0.4	92.5	97.2	87.0	91.8	96.4	85.0	91.3	96.1	84.0	91.7	96.2	82.0
	0.6	64.4	86.8	77.0	63.0	85.3	75.0	62.1	83.8	73.0	61.8	83.3	70.0
	0.8	11.6	21.0	53.0	11.8	20.7	49.0	11.6	19.9	42.0	11.5	19.8	34.0

Table 2: Accuracy results of synthetic experiments with Novelty Mention Model (NMM) data

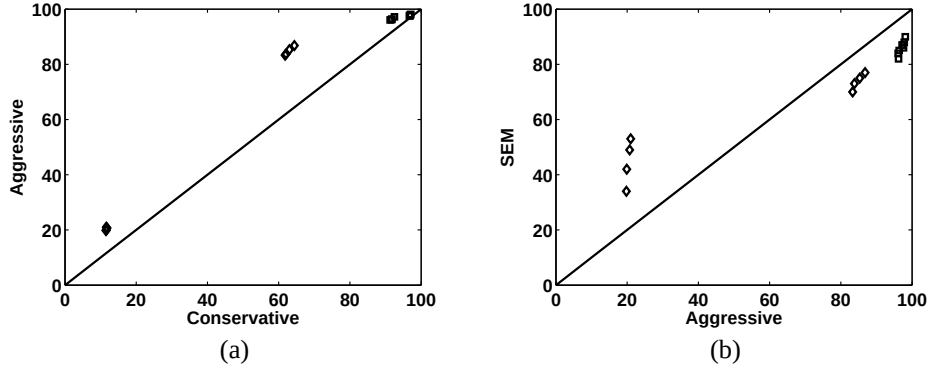


Figure 3: Accuracy for Novelty Mention Model data: (a) Aggressive vs. Conservative (b) SEM vs. Aggressive

the use of explicit mention models and their use in learning from radically incomplete and biased examples. Exploration of the relationship of this work to the statistical models of missing data is another fruitful direction.

References

- [Blum and Mitchell, 1998] Avrim Blum and Tom Mitchell. Combining Labeled and Unlabeled Data with Co-Training. In *COLT*, pages 92–100, 1998.
- [Carlson *et al.*, 2010] Andrew Carlson, Justin Betteridge, Richard C. Wang, Estevam R. Hruschka Jr., and Tom M. Mitchell. Coupled Semi-Supervised Learning for Information Extraction. In *WSDM*, pages 101–110, 2010.
- [Caruana, 1997] R. Caruana. Multitask Learning: A Knowledge-Based Source of Inductive Bias. *MLJ*, 28:41–75, 1997.
- [Doppa *et al.*, 2010] Janardhan Rao Doppa, Mohammad NasrEsfahani, Mohammad S. Sorower, Thomas G. Dietterich, Xiaoli Fern, and Prasad Tadepalli. Towards Learning Rules from Natural Texts. In *Workshop on Formalisms and Methodology for Learning by Reading (NAACL-2010)*, pages 70–77, 2010.
- [Friedman, 1998] Nir Friedman. The Bayesian Structural EM Algorithm. In *UAI*, pages 129–138, 1998.
- [Guha and Lenat, 1990] R. V. Guha and D. B. Lenat. Cyc: A Medterm Report. *AI Magazine*, 11(3), 1990.
- [Jaeger, 2006] M. Jaeger. The AI & M Procedure for Learning from Incomplete data. In *UAI*, pages 225–232, 2006.
- [Kharden and Roth, 1999] Roni Kharden and Dan Roth. Learning to Reason with a Restricted View. *Machine Learning*, 35(2):95–116, 1999.
- [Little and Rubin, 1987] R. J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data*. Wiley-Interscience, NY, 1987.
- [Michael and Valiant, 2008] Loizos Michael and Leslie G. Valiant. A First Experimental Demonstration of Massive Knowledge Infusion. In *KR*, pages 378–389, 2008.
- [Michael, 2009] Loizos Michael. Reading Between the Lines. In *IJCAI*, pages 1525–1530, 2009.
- [Nahm and Mooney, 2000] Un Yong Nahm and Raymond J. Mooney. A Mutually Beneficial Integration of Data Mining and Information Extraction. In *AAAI*, pages 627–632, July 2000.
- [Nijssen and Kok, 2003] Siegfried Nijssen and Joost N. Kok. Efficient Frequent Query Discovery in FARMER. In *PKDD*, pages 350–362, 2003.
- [Richardson and Domingos, 2006] Matthew Richardson and Pedro Domingos. Markov Logic Networks. *Machine Learning*, 62(1-2):107–136, 2006.
- [Schoenmackers *et al.*, 2010] Stefan Schoenmackers, Jesse Davis, Oren Etzioni, and Daniel S. Weld. Learning First-Order Horn Clauses from Web Text. In *EMNLP*, pages 1088–1098, 2010.
- [Yarowsky, 1995] D. Yarowsky. Unsupervised Word Sense Disambiguation rivaling Supervised Methods. In *ACL*, pages 189–196, 1995.

Textual Entailment Using Chain Clarifying Relationships

Octavian Popescu, Elena Cabrio and Bernardo Magnini

FBK-Irst (HLT Research Unit)

via Sommarive 18, 38123 Povo (Trento), Italy

popescu, cabrio, magnini@fbk.eu

Abstract

The Textual Entailment task requires a system to decide whether the information contained in a textual fragment (T) can be inferred from another text (H). Potentially, both fragments are ambiguous as each word may have different senses. In meaningful sentences only certain combinations of senses are legitimated, constraining the valid inferences that could be drawn. In this paper, we present a method to face TE by first reducing both texts to unambiguous intermediate forms, representing a unique combination of senses. The initial information is captured into particular types of relations between the phrases composing T and H respectively. We call these relationships Chain Clarifying Relationships (CCR). Entailment is decided by observing that *i*) the relation corresponding to T is the same or generalizes the relation in H; *ii*) the lexical items considered in the respective relations match.

1 Credits

The work described in the paper has been partially supported both by the Livememories project, and by the BCROCE project.

2 Introduction

Textual Entailment (TE) is a directional relation between a pair of text fragments, namely a *text* (T) and a *hypothesis* (H). The relation ($T \Rightarrow H$) holds whenever the truth of T follows from H. Defined as a task for automatic systems [Dagan and Glickman, 2004], TE requires to decide whether the information contained in H could be inferred from the information in T. The TE task aims at capturing major semantic inference needs in many Natural Language Processing applications, as question answering, summarization and document collection.

A broad scope of possible inferences is considered [Dagan *et al.*, 2009], including pairs with a high language variability (i.e. the same meaning is expressed by different words), and world knowledge. Being able to correctly detect the entailment relation for such pairs is difficult because a direct comparison between T and H at a string level is irrelevant. Furthermore, any sentence is potentially ambiguous, also due

to words polysemy. A pair cannot be correctly judged as entailment without selecting a particular reading for T and H respectively [Androutsopoulos and Malakasiotis, 2010].

For instance:

(1T) *A man was seen near the French bank.*

(1H) *A man was seen near a building.*

(2T) *A vessel anchored near the French bank.*

(2H) *A ship stationed near the French territory.*

Example (1) is a correct entailment pair only if *bank* is interpreted as referring to a building, but there is nothing in (1T) that imposes this reading. On the other hand, also pair (2) is a valid example of entailment, but in (2T) there is a strong coercion that forces the reading of *bank* as territory. Also, the reading of *vessel* as ship and *anchored* as stationed is enforced. Therefore the relationship between the constituents of (2T) is such that any of them is disambiguated, and a unique reading is possible. The only legitimate combination of senses for the individual words imposes that $2T \Rightarrow 2H$.

A reading (i.e. the interpretation of a sentence according to word senses) represents a class of sentences. For instance, from (3.1) to (3.3) the reading is the same, *see* means *watch*; from (3.4) to (3.6) the meaning of *see* is *become aware of*.

(3.1) *I see the drawings of the new generation.*

(3.2) *I see the pictures of John.*

(3.3) *I see the photos with John.*

(3.4) *I see the problem of the new generation.*

(3.5) *I see that there are issues with the pictures of John.*

(3.6) *I see what the photos of John mean.*

For instance, in (3.1) the lexical features associated with *drawings* disambiguate *see*, imposing the *watch* sense for it. On the other hand, the lexical features associated with *see* disambiguate *drawings*, imposing the sense *illustration* for it, and not the sense, let's say, *draining*. The relationship between these constituents is such that each one contributes in a chain like relationship to the disambiguation of the others. We call this relationship a *Chain Clarifying Relationship* (CCR). In a CCR each constituent is disambiguated, and only one combination of individual senses is possible. A CCR may characterize many contexts because it is expressed

in terms of a pattern containing lexical features. Examples (3.1)-(3.3) and examples (3.4)-(3.6) have the same CCR respectively. More specifically, we present a CCR as a pattern of generalized features that captures the relationship that allows the disambiguation of each constituent. Such pattern, called *Sense Discriminative* (SD) pattern, may characterize different contexts (i.e. textual fragments). If a context has the property that its components obey a CCR, then a unique reading is associated with the respective context. In this case, the corresponding SD pattern entails the respective context.

In this paper, we propose a method to detect the entailment relation between two contexts by *i)* associating a CCR, and consequently a SD pattern, to each of them, and *ii)* comparing the SD patterns to verify if there is a lexical match between the two contexts.

The paper is organized as follows. Section 3 reviews the relevant literature. Section 4 presents the processing steps from contexts to entailment relations via CCRs and SD patterns. Section 5 describes the results of a preliminary experiment we carried out to estimate the availability of data required by this approach. The paper concludes with Conclusion and Further work section.

3 Related Work

With respect to verb behaviour, selectional restrictions have been studied in word sense disambiguation [Resnik, 1997], [McCarthy *et al.*, 2001], [Briscoe, 2001].

[Lin *et al.*, 2001] transposed the distributional hypothesis from words to dependency trees, introducing the Extended Distributional Hypothesis. It states that paths in dependency trees have similar meanings if they tend to connect similar sets of words. In other words, two paths have high similarity if there is a large number of common features. However, not every feature is equally important, so it may be misleading. In [Kouylekov and Magnini, 2005] similarity metrics work on dependency trees. Contexts that are superficially different (e.g. active vs. passive) may have the same dependency tree and therefore can be compared.

The vectorial model has been employed for TE [Szpektor *et al.*, 2008]. Contexts are modelled as vectors and entailment rules can be applied only in similar contexts, where similarity is computed according to a similarity metric.

In [Mirkin *et al.*, 2009] textual units bigger than words are considered. The phrases that have not been seen in the training set are transformed into patterns that have already been seen. Their approach works on pairs with a high word similarity, while it is not directly applicable on the cases considered in this paper, namely contexts with little string similarity.

In [Pantel *et al.*, 2007] the context is represented as patterns, and rules that determine when such patterns can be used are induced. A problem of this approach is that the contexts are practically infinite. Possible solutions are *i)* to use a forest structure that allows multiple dependency trees to be processed as a single graph [Bar-Haim *et al.*, 2009]; *ii)* to generalize over such patterns, as in the approach we present here.

The CCR method we propose is complementary to many of these approaches.

4 CCRs in Textual Entailment

Consider the examples below:

(4.1) *He drove the girl to her father / the church.*

(4.2) *He drove the girl to craziness / to despair.*

Using a sense repository, such as WordNet, we can assign a sense to any of the words in both (4.1) and (4.2). In (4.1) the word *drive* has the sense *drive#3*, i.e. *cause someone or something to move by driving* and in (4.2) it has the sense *drive#5*, i.e. *to compel or force or urge relentlessly or exert coercive pressure on*. By comparing (4.1) and (4.2) and by consulting an ontology, we can identify a particular feature which characterizes the prepositional complements and which is responsible for the sense of *drive* in these sentences. The relationship between this feature and the sense of *drive* holds only in the common context of (4.1) and (4.2), i.e. the prepositional complement.

To each CCR corresponds a SD pattern that generalizes over all the contexts that present the same grammatical structure and that are composed by words having the same lexical features. To exemplify what a SD pattern looks like, we re-write the examples (3) and (4) in the format of CCRs:

(3.1)-(3.3)

SD= (subj=[Human] verb=see#1 obj=[ILLUSTRATION])

(3.4)-(3.6)

SD= (subj=[Human] verb=see#3 obj=[PROBLEM])

(4.1)

SD= (subj=[Human] verb=drive#3 obj=[HUMAN] prep=to objP=[LOCATION])

(4.2)

SD= (subj=* verb=drive#5 obj=[HUMAN] prep=to objP=[EMOTIONAL STATE])

The use of * shows that there is actually no restriction on the lexical feature for that slot.¹

Each CCR is associated uniquely with the inference its SD pattern represents. This property follows from the fact that in a CCR each word determines uniquely the sense of the other and there is only one way to understand the respective phrase. All the CCR contexts that have the same SD pattern entail the same sentence, namely exactly the sentence the SD represents. In example (5) we present a few sentences for which the same CCR holds and consequently, all the pairs [5.*T,5H] are valid entailments because 5H is the SD pattern corresponding to this particular CCR.

(5.1T) *I drive him to school.*

(5.2T) *He drives his assistant to his home.*

(5.3T) *He drove her mother to the mall.*

(5H) *A human drives a human to a location.*

¹There is no need to use * in the form of SD patterns, it has been used here for clarity reasons.

The sense of *drive* in (5) is *drive*#3. Changing the sense of the verb will change the CCR, which implies that another SD pattern is considered, and therefore that the initial entailment fails. Also, a prepositional complement with a different feature leads to a different CCR and consequently the initial entailment fails. The CCR property ensures the compositionality. A CCR entails the statements regarding the features of its components. For example:

- (3.2T) *I see the picture of John.*
 (3.2.1H) *There is a human.*
 (3.2.2H) *There is a human who sees#1.*
 (3.2.3H) *There is an illustration.*
- (5.1T) *I drive him to school.*
 (5.1.1H) *There is a human.*
 (5.1.2H) *There is a human who drives#3.*
 (5.1.3H) *There is a location.*

Entailment holds in the pairs [3.2T, 3.2.*H] and [5.1T, 5.1.*H]. The CCR property ensures that it is not possible that [3.2T] entails [3.2.2H], but does not entail [3.2.3H].

Since the SD pattern corresponding to a CCR uses features, instead of the actual words in the CCR, the lexical equality is not a constraint. Replacing the words in a CCR with different words that have the same lexical features does not affect the SD pattern. Therefore, the entailment relation is maintained in the pairs in which H is a sentence reproducing the SD pattern itself (see Example 6).

- (6T) *I stared at a picture of Mary.*
 (6H) *A human sees an illustration.*

The pair [6T, 6H] is a valid entailment because 6H is the SD pattern corresponding to the CCR expressed in 6T. From a practical point of view we are interested in pairs of sentences which are different lexical instantiations of the same SD pattern, like Example 7.

- (7T) *I see a picture of John.*
 (7H) *I stare at a picture of John.*

- (8T) *I stare at a picture of John.*
 (8H) *I see a picture of John.*

In the pair [7T, 7H] entailment holds, but in the pair [8T, 8H] it does not. The directionality of the entailment is given by the relations between the words considered in the SD pattern. The sentences in (7) and in (8) have the same SD pattern, and for this reason it is possible to compare them directly. Since lexically, *stare* is a hyponym of *see*, it means that entailment holds in Example 8, but not in Example 7. The relation of entailment is observed between two sentences *i*) which have the same SD pattern, and *ii*) when in H all the words in the CCR generalize the corresponding words in T. Predictably, if not all the words in H generalize the corresponding words of T there is no entailment. Indeed, for example, neither in [9.1, 9.2], nor in [9.3, 9.4] entailment holds. On the contrary, there is entailment in [9.4, 9.3] because each of the words in

(9.3) generalizes the words which are on the same positions in the SD pattern in (9.4).

- (9.1) *A mammal sees a picture of corn.*
 (9.2) *John sees a picture of a cereal.*
 (9.3) *A mammal sees a picture of a cereal.*
 (9.4) *John sees a picture of corn.*

If a sentence is not a CCR, then there are at least two possible readings. An ambiguous sentence has at least two different SD patterns which are partially overlapping. In this case, the entailment relation is non decidable.

- (10T) *I watched John's puppet.*
 (10.1H) *I saw John's puppet.*
 (10.2H) *I guarded John's puppet.*

The words in (10T) are not in a CCR, and, consequently the sentence is ambiguous. As there is no SD pattern associated with (10T) there cannot be an entailment relationship between [10T, 10.1H] or [10T, 10.2H].

We conclude this section summing up the connection between the CCRs and SD patterns, and the entailment relation:

- In a non ambiguous sentence the words are in CCR.
- A SD pattern represents a class of CCRs.
- A unique entailment is associated with a sentence for which a CCR holds. The sentence entails the SD pattern.
- Two sentences are in an entailment relationship iff:
 - they have the same SD pattern;
 - the sentence that is entailed (i.e. H) contains words which generalize the lexical features of the SD pattern of T.

5 Finding CCRs and Acquiring SD Patterns

The main issue to evaluate the feasibility of the proposed approach is to understand whether there are enough examples in available corpora, for allowing the correct extraction of the CCRs. In this Section we describe the preliminary experiment we carried out to test the availability of data needed for CCR construction. Since usually verbs are highly ambiguous, and enter into a large number of syntactic combinations, they are good candidates for our experiment.

In particular, we focused on a set of 30 verbs. Table 1 shows the results we obtained for a sample of them (column 1), namely those with the highest frequency in Semcor² (column 2, # occurrences). As the first step of our experiment, the part of context surrounding the target verb which is likely to form a CCR is manually identified. Then, as a second step, these contexts are generalized according to an ontology (SUMO). The most general form attained without having ambiguities is the final form of the pattern, i.e. the SD pattern.

While carrying out the first step, we decided to add some more examples for each verb, to obtain a complete list of CCRs. The underlying reason comes from the fact that the

²A subset of the English Brown corpus, sense-tagged according to Princeton WordNet 1.6. [Landes *et al.*, 1998]

verb	#occ	#add	#pat	#min	#max
begin	188	3	24	2	4
call	108	10	26	1	5
come	317	30	21	1	4
find	254	20	26	0	3
go	229	20	22	1	6
keep	166	20	14	0	4
leave	167	10	21	2	5
live	124	10	11	1	3
move	118	30	24	2	6
play	121	40	29	0	6
see	445	30	21	2	6
serve	112	10	14	1	3
use	291	40	21	2	5
work	120	30	24	1	6

Table 1: SD patterns in SemCor using SUMO

method we propose requires the identification of the minimal differences between two CCRs. These additional examples have the role of speeding up the process of SD patterns acquirement, and they are constructed basing on the sentences found in the corpus. Usually only one word is modified, i.e. it is replaced by a word with different lexical traits, which leads to a change in the meaning of the verb. If the corpus was well balanced with respect to the list of all verb senses, then the addition of new examples would not have been necessary. However, it seems that the best way to learn the discriminative patterns is to do it step by step, from simpler examples towards more complex ones (i.e. examples where the relevant information is fragmented by phrases that are less important in relation to sense changing). The number of additional examples for each verb (Table 1, column 3) has been empirically chosen: our estimation is that at least 2 or 3 examples for each sense are required to accurately identify a CCR.

Column 4, Table 1 lists the number of SD patterns obtained for each verb in the generalization process. The last two columns refer to the minimal and the maximal numbers of patterns extracted for each sense of the verb. Number 0 as minimum means that it was not possible to find different patterns for at least two senses of a verb.

6 Conclusion and future work

The paper presents a method to address the TE task by first reducing the textual fragments of a pair to unambiguous intermediate forms, representing a unique combination of senses. We introduced the notion of Chain Clarifying Relationships (CCR), as relations between the phrases in T and H that allow for the disambiguation of the constituents in the sentences, and for which only one combination of individual senses is possible. Such CCRs are represented as Sense Discriminative patterns (SD), i.e. classes of generalized contexts whose form is shared by pairs where entailment holds. A small scale experiment of acquisition of SD patterns has been carried out obtaining promising results, that suggests that a large scale acquisition of SD patterns is feasible. As for future work, we plan to focus on two directions: *i*) automatizing the CCRs discovery process, and *ii*) the application of this method to a large corpus of TE pairs (e.g. RTE data sets).

References

- [Androutsopoulos and Malakasiotis, 2010] I. Androutsopoulos and P. Malakasiotis. A survey of paraphrasing and textual entailment methods. *Journal of Artificial Intelligence Research*, 38:135–187, 2010.
- [Bar-Haim *et al.*, 2009] R. Bar-Haim, J. Berant, I. Dagan, I. Greental, S. Mirkin, E. Shnarch, and I. Szpektor. Efficient semantic deduction and approximate matching over compact parse forests. In *Proc. of the TAC 2008 Workshop on TE*, Gaithersburg, Maryland, USA., 2009.
- [Briscoe, 2001] T. Briscoe. From dictionary to corpus to self-organizing dictionary: Learning, valency associations in the face of variation and change. In *Proc. of Corpus Linguistics*, Lancaster University UK, 2001.
- [Dagan and Glickman, 2004] I. Dagan and O. Glickman. Probabilistic textual entailment: Generic applied modeling of language variability. In *Proc. of the PASCAL Workshop on Learning Methods for Text Understanding and Mining*, Grenoble, France, 2004.
- [Dagan *et al.*, 2009] I. Dagan, B. Dolan, B. Magnini, and D. Roth. Recognizing textual entailment: Rational, evaluation and approaches. *JNLE*, 15(Special Issue 04):i–xvii, 2009.
- [Kouylekov and Magnini, 2005] M. Kouylekov and B. Magnini. Tree edit distance for textual entailment. In *Proc. of RALNP-2005*, Borovets, Bulgaria., 2005.
- [Landes *et al.*, 1998] S. Landes, C. Leacock, and R.I. Teng. Building semantic concordances. In C. Fellbaum, editor, *WordNet: An Electronic Lexical Database*. The MIT Press, 1998.
- [Lin *et al.*, 2001] D. Lin, , and P. Pantel. Discovery of inference rules for question answering. *JNLE*, 7(4):343–360, 2001.
- [McCarthy *et al.*, 2001] D. McCarthy, J. Carroll, and J. Preiss. Disambiguating noun and verb senses using automatically acquired selectional preferences. In *Proc. of the SENSEVAL-2 Workshop at ACL/EACL’01*, Toulouse, France., 2001.
- [Mirkin *et al.*, 2009] S. Mirkin, L. Specia, N. Cancedda, I. Dagan, M. Dymetman, and I. Szpektor. Source-language entailment modeling for translating unknown terms. In *Proc. of ACL-IJCNLP*, Singapore., 2009.
- [Pantel *et al.*, 2007] P. Pantel, R. Bhagat, B. Coppola, T. Chklovski, and E. Hovy. Learning inferential selectional preferences. In *Proc. of NAACL HLT*, Rochester, NY., 2007.
- [Resnik, 1997] P. Resnik. Selectional preference and sense disambiguation. In *Proc. of ACL SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How?*, Washington, D.C., USA., 1997.
- [Szpektor *et al.*, 2008] I. Szpektor, I. Dagan, R. Bar-Haim, and J. Goldberger. Contextual preferences. In *Proc. of ACL: HLT*, Columbus, OH., 2008.

Learning Rules from Incomplete Examples via a Probabilistic Mention Model

Mohammad S. Sorower, Thomas G. Dietterich, Janardhan Rao Doppa, Prasad Tadepalli, and Xiaoli Fern

School of EECS, Oregon State University

Corvallis, OR 97331 USA

{sorower,tgd,doppa,tadepalli,xfern}@cs.orst.edu

Abstract

We consider the problem of learning rules from natural language text sources. These sources, such as news articles, journal articles, and web texts, are created by a writer to communicate information to a reader, where the writer and reader share substantial domain knowledge. Consequently, the texts tend to be concise and mention the minimum information necessary for the reader to draw the correct conclusions. We study the problem of learning domain knowledge from such concise texts, which is an instance of the general problem of learning in the presence of missing data. However, unlike standard approaches to missing data, in this setting we know that facts are more likely to be missing from the text in cases where the reader can infer them from the facts that are mentioned combined with the domain knowledge. Hence, we can explicitly model this “missingness” process and invert it via probabilistic inference to learn the underlying domain knowledge. This paper introduces an explicit *probabilistic mention model* that models the probability of facts being mentioned in the text based on what other facts have already been mentioned and domain knowledge in the form of Horn clause rules. Learning must simultaneously search the space of rules and learn the parameters of the mention model. We accomplish this via an application of Expectation Maximization within a Markov Logic framework. An experimental evaluation on synthetic and natural text data shows that the method can successfully learn accurate rules and apply them to new texts to make correct inferences.

1 Introduction

The immense volume of textual information available on the web provides an important opportunity and challenge for AI: Can we develop methods that can learn domain knowledge by reading natural texts such as news articles, journal articles, and web pages. We would like to acquire at least two kinds of domain knowledge: concrete facts and general rules. Concrete facts can be extracted as logical relations or as tuples to populate a data base. Systems such as Whirl [Cohen, 2000], TextRunner [Etzioni *et al.*, 2008], and NELL [Carlson

et al., 2010a] learn extraction patterns that can be applied to text to extract instances of relations.

General rules can be acquired in two ways. First, they may be stated explicitly in the text—particularly in tutorial texts. Second, they can be acquired by generalizing from the extracted concrete facts. In this paper, we focus on the latter setting: Given an incomplete data base of literals extracted from natural language texts, we seek to learn a set of probabilistic Horn clauses that capture general rules.

The problem of learning rules from extracted texts has been studied previously [Nahm and Mooney, 2000; Carlson *et al.*, 2010b; Schoenmackers *et al.*, 2010]. These systems rely on finding documents in which all of the facts participating in a rule are mentioned. If enough such documents can be found, then standard rule learning algorithms can be applied. A drawback of this approach is that it is difficult to learn rules unless there are many documents that provide such complete training examples. The central hypothesis of our work is that by explicitly modeling the process by which facts are mentioned, we can learn rules from sets of documents that are smaller and less complete.

To illustrate the challenges, consider the following sentence that discusses a National Football League (NFL) game:

“Given the commanding lead of Kansas city on the road, Denver Broncos’ 14-10 victory surprised many”

This mentions that Kansas City is the away team and that the Denver Broncos won the game, but does not mention that Kansas City lost the game or that the Denver Broncos was the home team. Of course these facts can be inferred from domain knowledge rules such as the rule that “if one team is the winner, the other is the loser (and vice versa)” and the rule “if one team is the home team, the other is the away team (and vice versa)”.

In our previous work [Doppa *et al.*, 2010], we employed an “implicit mention model” that did not explicitly represent the way in which mentions arise. Instead, it simply modified the way candidate rules were scored. The current paper builds on that work to show how an explicit mention model can be learned and applied.

2 Technical Approach

Consider a writer and a reader who share domain knowledge K . The writer wishes to efficiently communicate the conjunction of two (true) formulas $F \wedge G$.

Let $\text{READERWILLINFER}(G, F, K)$ denote the fact that the reader, when told F , will infer that G is true by applying the domain knowledge K . Then it suffices for the writer to mention only F and not G . We can express this as the general rule schema:

$$\text{MENTION}(F) \wedge \text{READERWILLINFER}(G, F, K) \wedge G \Rightarrow \neg \text{MENTION}(G).$$

Similarly, if the writer wishes to communicate the formula $F \wedge \neg G$, which is an exception to the general rule that $F \Rightarrow G$, then the writer must explicitly mention that G is not true:

$$\text{MENTION}(F) \wedge \text{READERWILLINFER}(G, F, K) \wedge \neg G \Rightarrow \text{MENTION}(\neg G).$$

In this paper, we consider only the first case. In addition, rather than explicitly modeling the reader's inference process, we instantiate the rule schema for each domain knowledge rule under the assumption that the reader's reasoning process can perform simple Horn clause chaining. Hence, given the rule $P \Rightarrow Q$, we define the mention rule

$$\text{MENTION}(P) \Rightarrow \neg \text{MENTION}(Q).$$

To represent and reason with the domain knowledge, the mention rules, and the observed mentions, we employ Markov Logic. There are four kinds of rules in our knowledge base (see Figure 1):

- **Fact-to-Fact Rules.** These are the unknown domain knowledge rules that we seek to learn. We prefix each literal with `FACT_` and write the rules in the form

$$\begin{aligned} &\text{FACT_HOMETEAM}(g, t2) \wedge \\ &\text{FACT_TEAMINGAME}(g, t1) \wedge \\ &\text{FACT_TEAMINGAME}(g, t2) \Rightarrow \text{FACT_AWAYTEAM}(g, t1). \end{aligned}$$

$\text{TEAMINGAME}(g, t)$ states that team t was one of the teams playing in game g . $\text{HOMETEAM}(g, t)$ states that team t was the home team in game g . Similarly, $\text{AWAYTEAM}(g, t)$ indicates that t was the away team in game g . (We apply the unique names constraint that variables with unique names must refer to distinct objects; hence $t1 \neq t2$ is assumed implicitly.)

- **Fact-to-Mention Rules.** These express the belief that if something is true, it is likely to be mentioned. These rules are probabilistic, and during learning they are assigned weights that reflect the baseline probability that any particular literal will be mentioned.

$$\text{FACT_HOMETEAM}(g, t) \Rightarrow \text{MENTION_HOMETEAM}(g, t).$$

- **Mention-to-Fact Rules.** These express the belief that the writer is not lying, so anything that is mentioned is true:

$$\text{MENTION_HOMETEAM}(g, t) \Rightarrow \text{FACT_HOMETEAM}(g, t).$$

Weights for these rules are not learned. Instead, the rules are assumed to hold with probability 1 (i.e., have infinite weight).

- **Mention-to-Mention Rules.** These rules instantiate the general mention rule schemas described above.

$$\begin{aligned} &\text{MENTION_HOMETEAM}(g, t2) \wedge \\ &\text{MENTION_TEAMINGAME}(g, t1) \wedge \\ &\text{MENTION_TEAMINGAME}(g, t2) \Rightarrow \\ &\quad \neg \text{MENTION_AWAYTEAM}(g, t1). \end{aligned}$$

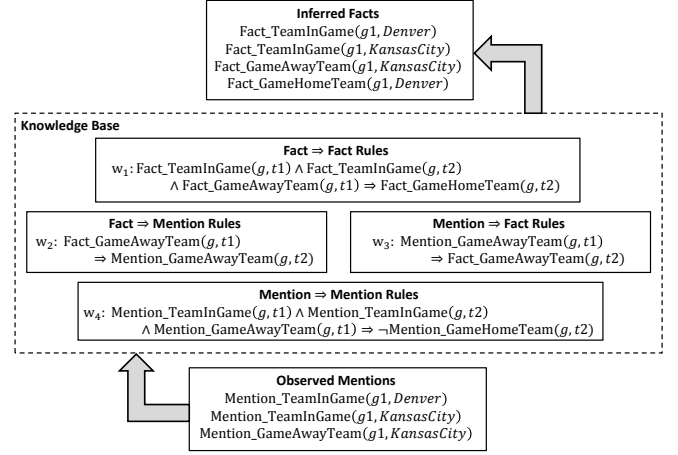


Figure 1: Markov Logic Model

In Markov Logic, each of these rules is assigned a (learned) weight which can be viewed as a cost of violating the rule. The probability of a world ω is proportional to

$$\exp \left(\sum_j w_j I[\text{Rule } j \text{ is satisfied by } \omega] \right),$$

where j iterates over all groundings of the Markov logic rules in world ω and $I[\phi]$ is 1 if ϕ is true and 0 otherwise.

An advantage of Markov Logic is that it allows us to define a probabilistic model even when there are contradictions and cycles in the logical rules. Hence, we can include both a rule that says “if the home team is mentioned, then the away team is not mentioned” and a rule that says “if the away team is mentioned, then the home team is not mentioned”. We can also include rules that say “the home team is mentioned” and “the away team is mentioned”. Obviously a possible world ω cannot satisfy all of these rules. The relative weights on these rules determine the probability that particular literals are actually mentioned.

Learning. We seek to learn both the rules and their weights. We proceed by first learning the fact-to-fact rules and then automatically generating the other rules (especially the mention-to-mention rules) from the general rule schema described above. Then we apply EM to learn the weights on all of the rules. This can have the effect of removing unnecessary rules by driving their weights to zero.

Learning Fact-to-Fact Rules. For each predicate, we generate a set of candidate Horn clauses with that predicate as the head. We consider all conjunctions of literals involving other predicates (i.e., we do not allow recursive rules) up to a fixed maximum length. Each candidate rule is scored on the training documents for *support* (number of training examples that satisfy the body) and *confidence* (the conditional probability that the head is true given that the body is satisfied). We discard all rules that do not achieve minimum support σ and then keep the top τ most confident rules. The values of σ and τ are determined via cross-validation within the training set. The selected rules are then entered into the knowledge base.

From each rule, we also derive a mention-to-mention rule that says if all of the literals in the body have been mentioned, then the head will not be mentioned. For each predicate, we also generate fact-to-mention and mention-to-fact rules.

Learning Weights. The goal of weight learning is to maximize the likelihood of the observed mentions (in the training set) by adjusting the weights of the rules. Because our training data only consists of mentions and no facts, the facts are latent (hidden variables), and we must apply the EM algorithm to learn the weights. Each game can be summarized by four literals: `HOME_TEAM(g, t1)`, `AWAY_TEAM(g, t2)`, `GAME_WINNER(g, t3)`, and `GAME_LOSER(g, t4)`. For purposes of weight learning, we must consider these four literals together, so we introduce an auxiliary predicate `GTUPLE(g, t1, t2, t3, t4)` and introduce the hard rule

$$\text{GTUPLE}(g, t1, t2, t3, t4) \equiv \text{HOME_TEAM}(g, t1) \wedge \text{AWAY_TEAM}(g, t2) \wedge \text{GAME_WINNER}(g, t3) \wedge \text{GAME_LOSER}(g, t4).$$

Given two teams, there are 16 possible instantiations of this literal in each game. Let us index them from 0 to 15 corresponding to `GTUPLE(g, A, A, A, A)` through `GTUPLE(g, B, B, B, B)`.

We employ the Markov Logic system Alchemy [Kok *et al.*, 2007] for learning and inference. In the E step, we apply the lazy MCSAT inference algorithm to estimate the marginal probability of each of the 16 possible instantiations of `GTUPLE`. In the M step, we then treat these possible instantiations as weighted evidence and apply generative learning to find the weight values that maximize the pseudo-likelihood of the weighted evidence. To introduce weighted evidence into Alchemy, we apply the following “dummy evidence trick”. Specifically, suppose we wish to introduce the evidence `GTUPLE(g1, A, B, A, B)` (configuration 5) with probability p_5 , for teams A and B . We define an evidence literal `GEVIDENCE(g1, A, B, A, B)` in Alchemy’s database file, and then add to the “mln” file a rule `GEVIDENCE(g1, A, B, A, B)  $\equiv$  GTUPLE(g1, A, B, A, B)` with a weight of

$$w_5 = \log p_5 - \frac{1}{K} \sum_{k=1}^K \log p_k,$$

where p_k is the marginal probability of configuration k . To avoid an explosion of instantiations during inference in Alchemy, we arbitrarily rename the teams in all of the games to be the constants A and B . This is a hand-coded form of lifted inference.

EM is iterated to convergence, which only requires a few iterations. Algorithm 1 summarizes the pseudo-code of the algorithm.

Treating Missing Mentions as Missing At Random: An alternative to the explicit mention model described above is to assume that the writer choose which facts to mention (or omit) at random according to some unknown probability. When data are missing-at-random (MAR), it is possible to obtain unbiased estimates of the true distribution via imputation using EM. We implemented this approach as follows. we apply the same method of learning rules (requiring minimum support σ and then taking the τ most confident rules).

Algorithm 1 Learn Explicit Mention Model

Input: $\mathcal{D}_I$ = Incomplete training examples

τ = number of rules per head

σ = minimum support per rule

Output: $\mathcal{M}$ = Explicit mention model

```

1: LEARN MENTION MODEL:
2: exhaustively learn rules for each head
3: discard rules with less than  $\sigma$  support
4: select the  $\tau$  most confident rules  $\mathcal{R}$  for each head
5:  $\mathcal{R}' := \mathcal{R}$ 
6: for each rule  $(factP \Rightarrow factQ) \in \mathcal{R}$  do
7:    $\mathcal{R}' := \mathcal{R}' \cup \{mentionP \Rightarrow \neg mentionQ\}$ 
8: end for
9: for every  $factP \in \mathcal{R}$  do
10:   add  $factP \Rightarrow mentionP$  to  $\mathcal{R}'$ 
11:   add  $mentionP \Rightarrow factP$  to  $\mathcal{R}'$ 
12: end for
13: repeat
14:   E-Step: apply inference to predict weighted facts  $\mathcal{F}$ 
15:   define complete weighted data  $\mathcal{D}_C := \mathcal{D}_I \cup \mathcal{F}$ 
16:   M-Step: learn weights for rules in  $\mathcal{R}'$  using data  $\mathcal{D}_C$ 
17: until convergence
18: return the set of weighted rules  $\mathcal{R}'$ 

```

Each learned rule has the general form `MENTION_A  $\Rightarrow$  MENTION_B`. The collection of rules is treated as a model of the joint distribution over the mentions (or, more precisely, over the possible groundings of `GTUPLE`). Generative weight learning is then applied to learn the weights on these rules. The marginal probabilities of `GTUPLE` are computed using the rule weights.

3 Experimental Evaluation

We evaluated our mention model approach on a synthetic data set to understand its behavior as we varied the amount of missing data. Then we compared its performance to the MAR approach on actual extractions from news stories about NFL football games.

Synthetic Data Experiment. The goal of this experiment was to evaluate the ability of our method to learn accurate rules from data that match the assumptions of the algorithm. We also sought to understand how performance varied as a function of the amount of information omitted from the text.

The synthetic data were generated using a database of NFL games (from 1998 and 2000-2005) downloaded from www.databasefootball.com. These data were then encoded using the following predicates: `TEAMINGAME(Game, Team)`, `GAMEWINNER(Game, Team)`, `GAMELOSER(Game, Team)`, `HOME_TEAM(Game, Team)`, `AWAY_TEAM(Game, Team)`, and `TEAM_GAMESCORE(Game, Team, Score)`, and treated as the ground truth. Note that these predicates can be divided into two correlated sets: $WL = \{\text{GAME_WINNER, GAME_LOSER, TEAM_GAMESCORE}\}$ and $HA = \{\text{HOME_TEAM, AWAY_TEAM}\}$.

From this ground truth, we generate a set of mentions for

Table 1: Synthetic Data Properties

q	0.17	0.33	0.50	0.67	0.83	0.97
Mentioned literals(%)	91.38	80.74	68.72	63.51	51.70	42.13
Complete records(%)	61.70	30.64	8.51	5.53	0.43	0.00

Table 2: Probabilistic Mention Model Performance on Synthetic Data. Each cell indicates the fraction of complete records inferred.

Training q	Test q					
	0.17	0.33	0.50	0.67	0.83	0.97
0.17	1.00	1.00	1.00	1.00	1.00	1.00
0.33	1.00	0.99	0.97	0.96	0.90	0.85
0.50	1.00	0.99	0.98	0.97	0.93	0.87
0.67	1.00	0.98	0.92	0.92	0.81	0.66
0.83	0.99	0.98	0.72	0.71	0.61	0.54
0.97	0.91	0.81	0.72	0.68	0.56	0.41

each game as follows. One literal is chosen uniformly at random from each of WL and HA and mentioned. Then each of the remaining literals is mentioned with probability $1 - q$, where q is a parameter that we varied in the experiments. Table 1 shows the average percentage of literals mentioned in each generated “news story” and the percentage of generated “news stories” that mentioned all literals. Note that the stories generated in this way reflect the actual statistics of game scores and outcomes.

Synthetic Experiments. For each q , we generated 5 different datasets, each containing 235 games. For each value of q , we ran the algorithm five times. In each iteration, one dataset was used for training, another for validation, and the remaining 3 for testing. The training and validation datasets shared the same value of q . The resulting learned rules were evaluated on the test sets for all of the different values of q . The validation set is employed to determine the thresholds τ and σ during rule learning and to decide when to terminate EM.

Table 2 reports the proportion of complete game records (i.e., all four literals) that were correctly inferred, averaged over the five runs. Note that any facts mentioned in the generated articles are automatically correctly inferred, so if no inference was performed at all, the results would match the second row of Table 1. Notice that when trained on data with low missingness (e.g. $q = 0.17$), the algorithm was able to learn rules that predict well over articles with much higher levels of missing values. This is because $q = 0.17$ means that only 8.62% of the literals are missing in the training dataset, which results in 61.70% complete records. These are sufficient to allow learning highly-accurate rules. However, as the proportion of missing literals in training data increases, the algorithm starts learning incorrect rules, so performance drops. In particular, when $q = 0.97$, the training documents contain *no* complete records (Table 1). Nonetheless, the learned rules

Table 3: Fraction of Literals Correctly Predicted for $q = 0.97$

Training q	Test q					
	0.17	0.33	0.50	0.67	0.83	0.97
0.97	0.98	0.95	0.93	0.92	0.89	0.85

are still able to completely and correctly reconstruct 41% of the games!

The rules learned under such high levels of missingness are not totally correct. Here is an example of one learned rule (for $q = 0.97$):

$$\text{FACT_HOMETEAM}(g, t1) \wedge \text{FACT_TEAMINGAME}(g, t1) \Rightarrow \text{FACT_GAMEWINNER}(g, t1).$$

This rule says that the home team always wins. When appropriately weighted in Markov Logic, this is a reasonable rule even though it is not perfectly correct (nor was it a rule that we applied during the synthetic data generation process).

In addition to measuring the fraction of entire games correctly inferred, we can obtain a more fine-grained assessment by measuring the fraction of individual literals correctly inferred. Table 3 shows this for the $q = 0.97$ training scenario. We can see that even when the test articles have $q = 0.97$ (which means only 42.13% of literals are mentioned), the learned rules are able to correctly infer 85% of the literals. By comparison, if the literals had been predicted independently at random, only 6.25% would be correctly predicted.

Experiments with Real Data: BBN provided the output of their extractor applied to two corpora of articles describing NFL games. The first corpus of articles is known as `BBN_training`. We will refer to it as D1. The second corpus is known as `BBN_robustness`. The articles in the second corpus were subjected to more thorough analysis, so the extractions are more complete. However, both D2 and (especially) D1 contain many errors including games involving more than two teams or where one team achieved multiple scores. These errors result primarily from coreference failures.

To address these problems, we decided to take each extracted game and apply a set of integrity constraints. The integrity constraints are learned from 5 complete game records. Examples of the learned constraints include “Every game has exactly two teams” and “Every game has exactly one winner.” Each BBN-extracted game is then converted into multiple games by deleting literals in all possible ways until all of the integrity constraints are satisfied. Finally, duplicate records are removed. After this processing, D1 contains 203 games and D2 contains 56 games. Table 4 summarizes the missingness of these game records.

The data from these repaired games is provided as training data to our algorithm. To measure the performance of the learned rules, we constructed a synthetic test set by taking 100 games from the downloaded NFL database and creating games in which various patterns of missingness were exhibited. Specifically, none of the test games mentioned both the home and away team or both the winning and losing team. In 20% of the test games, neither the home nor the away team was mentioned, whereas all test games mentioned exactly one

Table 4: Statistics on BBN-extracted and repaired games. “miss” means “missing”.

	Home/Away			Winner/Loser		
	miss both	miss one	miss none	miss both	miss one	miss none
D1	85.7	11.3	3.0	14.8	49.2	36.0
D2	17.9	58.9	23.2	17.9	57.1	25.0
Test	20.0	80.0	0.0	0.0	100.0	0.0

Table 5: Observed percentage of cases where exactly one literal is mentioned and the percentage predicted if the literals were missing at random

	Home/Away		Winner/Loser	
	observed	predicted	observed	predicted
D1	11.3	13.2	49.2	47.1
D2	58.9	49.9	57.1	49.8

of either the winner or the loser (see Table 4).

It is interesting to ask whether these data are consistent with the explicit mention model versus the missing-at-random model. Let us suppose that under MAR, the probability that a fact will be mentioned is p . Then the probability that both literals in a rule (e.g., home/away or winner/loser) will be mentioned is p^2 , the probability that both will be missing is $(1 - p)^2$, and the probability that exactly one will be mentioned is $2p(1 - p)$. We can fit the best value for p to the observed missingness rates to minimize the KL divergence between the predicted and observed distributions. If the explicit mention model is correct, then the MAR fit will underestimate the fraction of cases where exactly one literal is missing. Table 5 shows the results. For D1, the MAR model gives a fairly good fit to the observed missingness rates. However, for D2, it is clear that the MAR model seriously underestimates the probability that exactly one literal will be mentioned. Qualitatively, we believe that much of the missingness in D1 is due to extraction failures, whereas the missingness in D2 corresponds better to the assumptions of our probabilistic mention model.

We applied both our explicit mention model and the MAR model to these data sets. We measured performance relative to the performance that could be attained by a system

Table 6: Test set performance of our explicit mention model compared to the missing-at-random approach. Performance is measured as the number of games that are completely and correctly reconstructed by the learned models as a percentage of the number of games that can be completely reconstructed by the correct model.

Training Dataset	Explicit Mention Model	Missing at Random Model
D1	10.0	25.6
D2	100.0	6.0

that knows the correct rules. The results are summarized in Table 6. When trained on D1, the MAR method performs much better than our method, although both methods perform poorly. The challenge for both methods is that there are very few articles that provide complete training examples of the rules. When trained on D2, our method achieves perfect performance, whereas the MAR method only reconstructs 6% of the reconstructable games. This reflects the extreme difficulty of the test set, where none of the articles mentions all literals involved in any rule.

4 Conclusion

This paper has shown how to apply Markov Logic to represent a probabilistic model of the relationship between the true facts of a situation (such as an NFL game) and the propositions that are mentioned in an article about the situation. Experiments on synthetic data showed that the method is able to correctly reconstruct complete records even when neither the training data nor the test data contain complete records. On real data, the method achieves excellent performance when trained on the D2 dataset, whereas a method that treats facts as mentioned-at-random gives very poor performance. On dataset D1, performance was poor for both methods. This may reflect a fundamental lack of information sufficient to permit rule discovery, or it may reflect a weakness in our rule learning algorithm. It would be interesting to explore a modification of our algorithm in which candidate rules are scored within the Markov Logic mention model rather than relying on a separate rule-learning phase. This might permit the discovery of rules without observing *any* cases where both the head and the body are mentioned in a single document.

Acknowledgments

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. FA8750-09-C-0179. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the DARPA, the Air Force Research Laboratory (AFRL), or the US government.

References

- [Carlson *et al.*, 2010a] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E.R. Hruschka Jr., and T.M. Mitchell. Toward an architecture for never-ending language learning. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, pages 1306–1313. AAAI Press, 2010.
- [Carlson *et al.*, 2010b] Andrew Carlson, Justin Betteridge, Richard C. Wang, Estevam R. Hruschka, Jr., and Tom M. Mitchell. Coupled semi-supervised learning for information extraction. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining, WSDM '10*, pages 101–110, New York, NY, USA, 2010. ACM.
- [Cohen, 2000] William W. Cohen. WHIRL: A word-based information representation language. *Artificial Intelligence*, 118(1-2):163–196, 2000.

- [Doppa *et al.*, 2010] Janardhan Rao Doppa, Mohammad NasrEsfahani, Mohammad S. Sorower, Thomas G. Dietterich, Xiaoli Fern, and Prasad Tadepalli. Towards learning rules from natural texts. In *Proceedings of the NAACL HLT 2010 First International Workshop on Formalisms and Methodology for Learning by Reading*, FAM-LbR '10, pages 70–77, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.
- [Etzioni *et al.*, 2008] Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S. Weld. Open information extraction from the web. *Commun. ACM*, 51(12):68–74, 2008.
- [Kok *et al.*, 2007] S. Kok, M. Sumner, M. Richardson, P. Singla, H. Poon, D. Lowd, and P. Domingos. The Alchemy system for statistical relational AI. *Technical report, Department of Computer Science and Engineering, University of Washington, Seattle, WA*, 2007.
- [Nahm and Mooney, 2000] Un Yong Nahm and Raymond J. Mooney. A mutually beneficial integration of data mining and information extraction. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and the Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 627–632. AAAI Press, 2000.
- [Schoenmackers *et al.*, 2010] Stefan Schoenmackers, Oren Etzioni, Daniel S. Weld, and Jesse Davis. Learning first-order Horn clauses from web text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 1088–1098, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

Evaluating Correctness of Propositions Using the Web

Mehdi Samadi and Manuela Veloso and Manuel Blum

Carnegie Mellon University

{msamadi,veloso,mblum}@cs.cmu.edu

Abstract

In this work, we contribute a method that takes advantage of the powerful corpus of the Web data to automatically evaluate the truth of propositions that are stated as multi-argument instantiated predicates, e.g., *City_In_Country(Beijing,China)*. Our approach, OpenEval, automatically converts a given instantiated predicate into a Web search query, then extracts a corresponding set of features from the web pages returned. Initially, OpenEval trains a classifier on a list of predicates by using a set of seed positive examples for each predicate. Each such set furthermore provides negative examples for the other predicates. To evaluate a new query, OpenEval again converts the query into a corresponding set of features extracted from the Web. The extracted features are then used as input to the learned classifier. The classifier output is used to calculate the correctness probability of the input predicate. We experimentally show that OpenEval is significantly superior to the previous related techniques, in particular the Pointwise Mutual Information (PMI) and Never-Ending Language Learner (NELL).

1 Introduction

“Is yogurt healthy?”, “Is zero a number?”, “Is pumpkin a fruit?”, these are a few examples of the most popular questions that have been searched in Google.¹ These questions can be converted to propositions that require assessment of their correct value. In general, evaluating correctness of a proposition is of interest to information processing applications, such as automated Question Answering (QA). Most of the questions and their candidate answers can be converted to a proposition that requires assessment. For example the question “where is Beijing located?” and the corresponding answer “China” can be converted to a proposition “Beijing is located in China”. It has been shown that the accuracy of a QA system can be improved by using automated answer validation techniques [4] that assess the correctness of the answers found by the QA system.

Evaluating correctness of a proposition can also be used in Information Extraction (IE) systems to assess the probability that information is extracted correctly [3]. IE systems such as KnowItAll [3], TextRunner [2] and NELL [1] extract a large set of facts from their input text corpus. Etzioni et al. showed [3] that the accuracy of IE systems can be improved by validating the correctness of the extracted facts. In particular, they used search engine hit counts to

assign the probability of correctness to the facts that are extracted by their KnowItAll system.

This paper presents *OpenEval*, a new IE approach that is able to **learn** how to **evaluate** correctness of a proposition. OpenEval uses multi-argument predicates such as *City_In_Country(Beijing,China)* [‘Is Beijing a city in China?’] to represent the extracted information. A predicate such as $p(x_1, \dots, x_n)$ defines a relationship between entities $(x_1, \dots, x_n)$. We call $(x_1, \dots, x_n)$ an *instance* of predicate p . As part of training, OpenEval is provided a set of predicates P (e.g., *City(x)*, *City_In_Country(x,y)*) and a few seed examples for each predicate $p \in P$ (e.g., *Los Angeles* and *New York* for predicate *City*). For each predicate $p \in P$ and i_p , an instance of p , OpenEval first automatically converts i_p into a Web search query and then extracts the corresponding set of features from the Webpages returned by the search engine. Each of the extracted features consists of a single word or a combination of a set of words (e.g., *located-in*). The extracted features are then used to train a classifier (e.g., SVM). The goal of the classifier is to classify each set of the input features to one of the predicates in P .

To validate correctness of an input predicate instance, OpenEval converts the input predicate instance to a search query and extracts a corresponding set of features from the Web. The extracted features are then used as input to the learned classifier. The classifier output is used to calculate the correctness probability of the input predicate.

The baseline for our work is the Pointwise Mutual Information (PMI) technique, that has been widely used in IE [5; 6; 3], and Never-Ending Language Learner(NELL) [1]. We train OpenEval on a set of mutually exclusive predicates that are chosen randomly from 400 predicates in NELL’s ontology. We experimentally show that OpenEval improves F1 measure on the test data at least by a factor of two compared to PMI and NELL techniques.

2 Related Work

Researchers have worked on the problem of evaluating correctness of a proposition from both information extraction and question answering points of view. Turney [6] presented an unsupervised technique that uses search engine hit counts to measure the similarity of pairs of words. They also used the PMI technique to measure semantic similarity between two words. Other researchers have used PMI to validate information extraction [5] or to validate a candidate answer in a question answering system [4]. Soderland et al. [5] showed different methods of using PMI which can be used to assign a correctness probability to the extracted facts.

The performance of all the above techniques depends on the accuracy of the search engine hit counts. However, the search engine hit counts are only a crude estimate of the number of matching documents and it has been shown that hit count estimates are not accurate [7]. However, our approach uses the content of the webpages

¹Similar popular questions can be obtained by entering words, such as “Is” or “Does” in Google, and seeing Google’s suggestions.

New York is a vibrant and eclectic place. There is an incredibly diverse range of things to see and do- but there are a few attractions like no other. No visit to **New York City** is complete without visiting the Empire State Building, the **Statue of Liberty** and the Ellis Island Immigration Museum. Ellis Island is where it all began. You will receive a free ferry ride to the Island and the **Statue of Liberty** with the **New York Pass**.

Figure 1: An example of the text that is found for predicate *City_Attraction(Statue of Liberty, New York)*.

that are returned by the search engines for the evaluation and does not rely on the search hit counts.

NELL (Never-Ending Language Learning) [1] is a semi-supervised learning that has been developed as part of the Read the Web (RTW) project at Carnegie Mellon University. It extracts structured information from the unstructured web pages. The input of NELL is an initial ontology (e.g. hundreds of predicates with one or two arguments) and 10 to 15 seed examples for each predicate that is defined in its ontology. Given this input, NELL extracts new instances of predicates from a collection of 500 million webpages. Given an instance of predicate p , we can simply validate it by checking if the input instance is in the NELL’s Knowledge Base (KB). In our experimental result, we compare result of OpenEval to NELL.

3 OpenEval

OpenEval considers a set P of predicates p with optional associated keywords K_p for each p , and instantiation i_p for each predicate. For example, for $p = \text{City}$, values of i_p are *New York* and *Beijing*, and for $p = \text{City_Country}$, values of i_p are (*New York, USA*) and (*Beijing, China*). OpenEval evaluates the correctness of a combined tuple $t = \langle p, i_p \rangle$, using the keywords K_p , if any.

We describe next in detail the components of OpenEval. Throughout the presentation, we exemplify with a set of predicates $P = \{\text{City_Attraction}, \text{Country_Currency}\}$ and a set of corresponding instantiations as two subsets of seed examples of attractions of cities and currencies of varied countries. We further use a set of keywords $K_{ca} = \{\text{popular}, \text{unique}\}$ for *City_Attraction*, and $K_{cc} = \emptyset$ (empty set) for *Country_Currency*.

3.1 Feature Extractor

The input to the *feature extractor* is the tuple $t = \langle p, i_p, K_p, N \rangle$, where $p \in P$ is the input predicate, i_p is an instance of predicate p , $K_p \in K$ is a set of optional keywords, and N is the number of webpages that the feature extractor should extract feature from. The output of the feature extractor is a set of *bag-of-features*, denoted by B_{i_p} . Each $b \in B_{i_p}$ is a bag-of-features that is extracted for the input instance i_p . Each *feature* is a single word or a combination of a set of words (e.g., *located-in*). A *bag-of-features* contains a set of features (there may be multiple instances of a feature) that are extracted by the feature extractor. For example, $\{\text{must-See}, \text{maps}, \text{nearby-restaurants}, \text{Admission}, \text{tickets}, \text{prices}, \text{entry}\}$ is part of one of the bag-of-features that is extracted for the predicate *City_Attraction*. In our explanation, B_{i_p} indicates a set of bag-of-features that are extracted for instance $i_p \in I_p$.

Algorithm 1 shows the feature extraction procedure. Given the input tuple $t = \langle p, i_p, K_p, N \rangle$, the feature extractor first builds the query Q (Lines 3-4). The query Q is built from the name of the predicate p , arguments of the input instance i_p , and the keywords K_p . For example, suppose that (*Statue of Liberty*, *New York*) is given as an instance of the predicate *City_Attraction*. The query Q

Algorithm 1 Feature Extractor

Input: $t = \langle p, i_p, K_p, N \rangle \leftarrow$ Input Tuple

- 1: **Function:** *FeatureExtractor* ($T = \langle p, i_p, K_p, N \rangle$)
- 2: $B_{i_p} \leftarrow \phi$ // B_{i_p} is the set of all the bag-of-features for i_p
- 3: $pArgs \leftarrow$ Arguments of i_p separated by space
- 4: $Q \leftarrow "p \ pArgs \ K_p"$ // Q is the search query
- 5: $W \leftarrow$ Get the first N results of Google for query Q
- 6: $B_{W_i} \leftarrow \phi$ // B_{W_i} is the list of all the bag-of-features that are extracted from webpage W_i
- 7: **for all** Webpages $W_i \in W$ **do**
- 8: **for all** occurrences p_j of $pArgs$ in W_i **do**
- 9: $\bar{T}_j \leftarrow$ extract text around $pArgs$
- 10: $b \leftarrow$ all the words and bigrams in $\bar{T}_j$
- 11: Add b to B_{W_i}
- 12: **end for**
- 13: Merge all the bag-of-features in B_{W_i} and add to B_{i_p}
- 14: **end for**
- 15: **return** B_{i_p}

would then be $\{\text{"Statue of Liberty"} \text{"New York"} \text{popular unique}\}$. The feature extractor then searches the query Q in Google and downloads the first N webpages (Line 5). For each webpage W_i that is found by Google, the feature extractor searches the content of W_i and extracts all the features in W_i . The feature extractor finds all the positions p_j in W_i whose words in $pArgs$ appear “close” to each other (Line 8). “Close” means we allow words in $pArgs$ to appear in any arbitrary order and up to a maximum of 10 words can be in between, before, and after them. For each position p_j , consider $\bar{T}_j$ as the text that occurs around $pArgs$. The bag-of-features b is then built from all the words and bigrams in $\bar{T}_j$. All the stop words and also the bigrams that contain two stop words are deleted from b . All the bag-of-features b that are found in webpage W_i are added to B_{W_i} (Line 11). Figure 3.1 shows an example of the text that is found in one of the returned webpages for query Q . The arguments of the predicate *City_Attraction* are shown in bold. Only in two places a maximum of 10 words occur between words in $pArgs$. The underlined text shows the value of $\bar{T}_j$ for each occurrence of words in $pArgs$. The feature extractor returns two bag-of-features in this case: $b_1 = \{\text{attractions}, \text{few-attractions}, \text{attractions-like}, \text{visit}, \text{No-visit}, \text{visit-to}, \dots\}$, and $b_2 = \{\text{receive}, \text{receive-a}, \text{free}, \text{a-free}, \text{free-berry}, \text{berry}, \dots\}$.

The final step of the algorithm is to build the set of bag-of-features B_{i_p} from each element of B_{W_i} (Lines 13). To do this, all the features that are extracted from the same webpage W_i , are merged to a single bag-of-features. In this case, only one single bag-of-features is built for each $W_i \in W$. The extracted bag-of-features contains all the words that appear around words in $pArgs$ at different positions of the webpage W_i . For example, b_1 and b_2 are merged to a single bag-of-features: $v = \{\text{attractions}, \text{few-attractions}, \text{attractions-like}, \text{visit}, \text{No-visit}, \text{visit-to}, \text{receive}, \text{receive-a}, \text{free}, \text{a-free}, \text{free-berry}, \text{berry}, \dots\}$.

3.2 Training Data Constructor

The role of the *training data constructor* is to build a set of training data that are later used to train a classifier. The algorithm takes the tuple $T = \langle P, I, K, N \rangle$ as an input. For each predicate $p \in P$ and each instance $i_p \in I_p$, the feature extractor is called by given the tuple $\langle p, i_p, K_p, N \rangle$ as its input. The feature extractor extracts and returns a set of bag-of-features B_{i_p} that are extracted for i_p . Each of the bag-of-features $b \in B_{i_p}$ is used as one of the training examples. The label for b is the same as the name of predicate p .

3.3 Classifier

The next step is to train a *classifier* on a set of bag-of-features B that is built by the *training data constructor*. First, each of the bag-of-features $b \in B$ is transformed into a *feature vector* f , where each element corresponds to the frequency of a distinct feature in b . The dimension of the vector is equal to the total number of distinct features (i.e., words and bigrams) that exist in B . The label for the feature vector f is the same as the label of the input bag-of-features b . We train a classifier such as SVM to classify the constructed feature vectors to different classes. Each class represents a predicate $p \in P$.

The trained classifier is later used in the evaluation part of OpenEval to classify the input bag-of-features b_{i_p} to one of the predicates $p \in P$. In this case, b_{i_p} is first transformed to a feature vector and then is given to the trained classifier. The classifier classifies the input feature vector to one of the predicates $p \in P$.

3.4 Evaluator

The input of the *evaluator* is a tuple $t = \langle p, i_p, K_p, N \rangle$, where p is a predicate $p \in P$, i_p is a candidate instance of predicate p that the evaluator is required to make an assessment, K_p is a set of keywords for predicate p , and N is the number of webpages that evaluator should process. The input keywords K_p should be the same as the set of keywords that are used in the training. The evaluator first calls the feature extractor to extract a set of bag-of-features for the input tuple t . The feature extractor returns B_{i_p} which contains a set of bag-of-features related to the predicate i_p . For each $b_{i_p} \in B_{i_p}$, the trained classifier is then called to classify b_{i_p} to one of the predicates in P .

The correctness probability of the input predicate instance i_p is calculated by dividing the number of times that $b \in B_{i_p}$ is classified as p to the size of B_{i_p} . For example, suppose that *City Attraction* and (*Statue of Liberty, New York*) are given as p and i_p . Also suppose that feature extractor extracts features from 8 webpages (i.e., $N = 8$). In this case, B_{i_p} contains the maximum of eight bag-of-features, each of which is a set of features that are extracted from the same website. Each $b \in B_{i_p}$ is given to the classifier that is trained in the classification part. The classifier classifies b to one of the predicates $p \in P$. The correctness probability is obtained by the number of times that b is classified as *City Attraction* divided by the total number of bag-of-features in B_{i_p} (maximum is eight).

To decide if i_p is an instance of predicate p , one can check if the correctness probability that is found by the evaluator is greater than a predefined threshold. Given a set of seed examples as an input, the process of choosing a threshold Tr_p for each predicate p is as follows. First, for each predicate $p \in P$, the seed examples are split into two sets, T_1 and T_2 . OpenEval is then trained using all the seed examples in T_1 . Now in order to set the value of Tr_p , we calculate the F1 score for all integers between 0 and 100 using the seed examples in set T_2 , and the integer which achieves the best F1 score is assigned to Tr_p .

4 Experimental Evaluation

4.1 Setup

OpenEval is tested on two sets of predicates, one set contains predicates with one argument (i.e., categories) and the other contains predicates with two arguments (i.e., relations) that are chosen randomly from 400 predicates of NELL's knowledge base.² Our approach is general and can be used for predicates with any number of arguments. However to be able to compare our results with other

²The predicates are chosen randomly under the condition that they should be mutually exclusive.

related work, we have tested OpenEval only on the predicates with one and two arguments. Table 1 lists all of the categories in the leftmost column, and Table 2 lists all the relations in the leftmost column. For each predicate, 15-25 positive seed examples are provided as an input to the OpenEval. OpenEval is trained and tested separately on each set of the predicates with one and two arguments. We have used Support Vector Machines (SVM) as the classification technique in OpenEval.

To evaluate OpenEval, 30 random positive instances are chosen for each of the input predicates from *Wikipedia.org*. Wikipedia contains a set of webpages that have listed instances of different categories and relations (e.g., http://en.wikipedia.org/wiki/List_of_academic_disciplines). Each positive test example for one of the predicates is considered as a negative test example for the other predicates since all the predicates are mutually exclusive.

We have compared our technique to both NELL and PMI using standard performance metrics of precision, recall, and F1. NELL extracts instances of categories and relations that are defined in its ontology from a large Web corpus that contains 500 million webpages (the ClueWeb09 corpus) [1]. Given an instance i_p of predicate p , we can simply use NELL to check if i_p is an instance of p . This can be done by checking if i_p exists in the NELL's Knowledge Base (KB) as an instance of predicate p . We have used NELL's KB that is built from 196 iterations. Another baseline for our comparison is PMI technique. Given a predicate p and a predicate instance i_p , the PMI score is calculated as described in [6]. The thresholds for both PMI and OpenEval techniques are calculated using the the same technique that is explained in Section 3.4.

4.2 Experimental Results

Tables 1 and 2 show the Precision (Pr), Recall (Re) and F1 score (F1) of PMI, NELL, and OpenEval for predicates with one argument (i.e., categories) and predicates with two arguments (i.e., relations). The predicates are randomly chosen from NELL's ontology. The experiments are obtained when OpenEval uses 56 webpages (i.e., $N = 56$) for the training and 8 webpages (i.e., $N = 8$) for the evaluation. The first column shows the predicates that are used in the experiments. Across all the predicates, we can see that OpenEval has achieved the best average F1 score (48% for categories and 45% for relations).

Comparing the result of NELL and OpenEval, it can be seen that performance of both NELL and OpenEval are almost the same for the categories. NELL and OpenEval have achieved average F1 score of 43% and 48% while the F1 score of PMI is 13%. NELL's accuracy for relations is significantly less compared to categories. Carlson et al. [1] have also stated that NELL's performance is significantly better for categories compared to relations. Comparing the average F1 score of OpenEval, NELL, and PMI on relations, it can be seen that accuracy of OpenEval is about 2.5 times higher than both NELL and PMI.

5 Conclusion

This paper introduced OpenEval, a novel information extraction approach that is able to automatically evaluate correctness of a predicate instance. OpenEval can be used to validate information that extracted by IE systems or to validate a candidate answer that is extracted by a QA system. We described two main components of OpenEval in detail: *learning* and *evaluation*, and experimentally showed that OpenEval massively outperforms the PMI technique. We also compared the result of our system to NELL's KB. Comparison of the average F1 score of NELL and OpenEval shows that OpenEval outperforms NELL on predicates with two arguments at least by a factor of six. Both NELL and OpenEval have achieved

Predicate Name	PMI			NELL			OpenEval		
	Pr(%)	Re(%)	F1(%)	Pr(%)	Re(%)	F1(%)	Pr(%)	Re(%)	F1(%)
<i>Academic Field</i>	14	37	21	100	20	33	26	47	33
<i>Animal</i>	5	100	9	100	77	87	22	57	32
<i>Board Game</i>	5	100	9	86	20	32	49	60	54
<i>Emotion</i>	12	70	20	97	93	95	19	67	29
<i>Furniture</i>	8	13	10	100	60	75	38	50	43
<i>Geometric Shape</i>	5	100	9	100	17	29	63	50	56
<i>Sports Equipment</i>	31	13	19	100	5	9	43	20	27
<i>Ethnic Group</i>	10	13	12	100	17	29	23	73	35
<i>ML Algorithm</i>	14	13	14	100	30	46	33	33	33
<i>Religion</i>	5	100	9	100	27	42	66	46	54
<i>Protein</i>	100	13	24	93	47	62	85	77	81
<i>Programming Language</i>	5	100	9	100	5	9	16	57	25
<i>Musician</i>	71	17	27	100	13	24	61	73	67
<i>Scientist</i>	5	100	9	100	5	9	29	63	40
<i>Park</i>	5	100	9	100	27	42	36	67	47
<i>Mountain</i>	5	100	9	100	47	64	53	60	56
<i>Food</i>	5	100	9	100	7	13	62	77	69
<i>Film Festival</i>	5	100	9	100	40	57	69	97	81
<i>Building Material</i>	12	17	14	100	5	9	33	50	39
<i>Disease</i>	67	7	12	100	50	67	50	60	55
<i>Model</i>	5	100	9	100	50	67	57	67	62
Average	19	63	13	99	31	43	44	60	48

Table 1: The performance (F1, precision, recall) of PMI, NELL, and OpenEval techniques for predicates with one argument.

Predicate Name	PMI			NELL			OpenEval		
	Pr(%)	Re(%)	F1(%)	Pr(%)	Re(%)	F1(%)	Pr(%)	Re(%)	F1(%)
<i>Language of Country</i>	31	13	19	100	33	50	29	53	38
<i>Musician Plays Instrument</i>	7	80	13	100	3	6	39	30	34
<i>Company's Office In City</i>	6	93	11	0	0	0	13	13	13
<i>Country Currency</i>	21	23	22	0	0	0	70	70	70
<i>Team Plays Sport</i>	10	83	17	100	40	57	32	73	44
<i>Company Produces Product</i>	8	53	15	0	0	0	30	50	38
<i>Sport Uses Equipment</i>	9	66	16	0	0	0	46	45	46
<i>Airport In City</i>	6	100	12	100	16	28	71	48	58
<i>Politician US Holds Office</i>	6	100	11	0	0	0	33	40	36
<i>Park In City</i>	6	93	12	66	7	13	63	40	49
<i>Stadium Located In City</i>	6	80	11	100	10	18	38	57	45
<i>Museum In City</i>	6	100	12	100	3	6	75	50	60
<i>Aquarium In City</i>	7	83	13	0	0	0	45	43	44
<i>Person Born In City</i>	7	77	12	0	0	0	45	17	24
<i>Team Plays In League</i>	8	93	15	100	100	100	54	47	50
<i>Body part Contains Body part</i>	7	90	12	0	0	0	58	47	52
<i>Drug Has Side Effect</i>	6	100	11	100	13	23	64	53	58
Average	9	78	14	51	13	18	47	46	45

Table 2: The performance (F1, precision, recall) of PMI, NELL, and OpenEval techniques for predicates with two arguments.

almost the same results on predicates with one argument. OpenEval is a general technique, easy to implement, and has relatively low training time.

References

- [1] Andrew Carlson, Justin Betteridge, Richard C. Wang, Estevam R. Hruschka Jr., and Tom M. Mitchell. Coupled semi-supervised learning for information extraction. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining (WSDM 2010)*, 2010.
- [2] Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S. Weld. Open information extraction from the web. *Commun. ACM*, 51:68–74, December 2008.
- [3] Oren Etzioni, Michael Cafarella, Doug Downey, Stanley Kok, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. Web-scale information extraction in knowitall: (preliminary results). In *Proceedings of the 13th international conference on World Wide Web, WWW '04*, pages 100–110, New York, NY, USA, 2004. ACM.
- [4] Bernardo Magnini, Matteo Negri, Roberto Prevete, and Hristo Tanev. Is it the right answer? exploiting web redundancy for answer validation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 425–432, 2002.
- [5] Stephen Soderl, Oren Etzioni, Tal Shaked, and Daniel S. Weld. The use of web-based statistics to validate information extraction. In *AAAI Workshop on Adaptive Text Extraction and Mining*.
- [6] Peter Turney. Mining the web for synonyms: Pmi-ir versus lsa on toefl, 2001.
- [7] Ahmet Uyar. Investigation of the accuracy of search engine hit counts. *J. Inf. Sci.*, 35:469–480, August 2009.

Learning Causality from Textual Data

Kira Radinsky*, Sagie Davidovich[†] and Shaul Markovitch*

Computer Science Department

Technion–Israel Institute of Technology

*{kirar, shaulm}@cs.technion.ac.il, [†]{mesagie}@gmail.com

Abstract

We present a new methodology for modeling and predicting future events through machine learning and data mining techniques from textual data. Modeled events span across varied domains including politics, economy and society. The model employs human-style prediction techniques such as causality inference, generalization and projection based on past experience. For this purpose, we use news archives that date back 150 years as a vast source of text representing “past-experiences” and inference patterns. Empirical evaluation on real news articles shows that the ability of our algorithm to predict future events is similar to that of humans.

1 Introduction

Predicting events in politics, economics, society, etc. is an intriguing task that is usually performed by human experts possessing extensive domain-specific and common-sense knowledge. Much of the causal knowledge that helps humans understand the world is found in texts, that expresses people’s beliefs. Understanding causality and the ability to predict are fundamental capabilities of intelligent behavior and are essential for decision making and other human common-sense reasoning, such as question-answering.

Psychological studies [Kahneman and Tversky, 1973] provide evidence that humans are good at certain types of event predictions – especially as they base on their lifetime knowledge about how the world behaves. They possess wide common-sense knowledge about how the world is now, how it was before, what happens after an action, and which event causes another.

Can we endow a machine with such rich capabilities to allow it to predict events? Specifically, can we obtain all this expertise which humans are exposed to throughout their lifetime experience? The World Wide Web encapsulates much of our humans knowledge. It treasures information about historical events through news archives and encyclopedias. It has dynamic updates about current events through online news papers updating every

minute. On top of all the past and current human history, the web has many common sense ontologies that can be utilized to extract causality patterns and generalize events. This knowledge can serve as the basis for performing true human-like prediction – with the ability to learn, understand language, possess intuitions and general world knowledge.

Although some works dealt with entity and relation extraction [Carlson *et al.*, 2010], including causal extraction specifically [Chan and Lam, 2005], and temporal information extraction [Ling and Weld, 2010] from the web, to the best of our knowledge, none have dealt with causality prediction. Other related works deal with information overflow – a somehow related work is [Shahaf and Guestrin, 2010] which, given two news articles, provides a coherent small number of news items that connects them. In our work we receive an event and predict its effect.

Time and time again we can draw parallels from what is happening today, to historical examples in the past. And although it is impossible for history to repeat itself exactly, there are indeed recurring themes and patterns, in the historical time line of mankind. Therefore, finding similar occurrences in the past, and observing what they caused, might give us great insight on the present. Interestingly, psychological studies ([Kahneman and Tversky, 1973]) show that people also predict by similarity. We present a new methodology for predicting events using a learning algorithm, which given an event represented in natural language, predicts a future event it can cause. We first introduce a method for event representation inspired by Kim’s (1993) property exemplification of events theory and Schank’s (1972) conceptual dependency theory. We then present a prediction algorithm, that uses world knowledge to generalize the events it was trained on in the past. During prediction, when presented with a never-seen-before event, it matches the event to a similar generalized or specific event in the past. It then projects the present event to a predicted future event, using a cause-effect clause pattern it learnt during training. For example, given a present event “elections in Tehran”, and a matched past causality of “elections in Baghdad” caused “protests in Iraq”, the algorithm will output “protests in Iran” rather than the naive output

of “protests in Iraq”.

For training the algorithm, we have created a graph of 300 millions fact nodes connected by more than one billion edges. Temporal data expressing causality was obtained mainly from the New-York-Times archives (dating back to 1851). We applied semantic natural language modeling techniques on the textual data, creating a structured representation of the knowledge.

We present an empirical framework for evaluating the performance of the algorithms. Given a set of events extracted from the 2010 news, humans were asked to indicate what they believe would happen as a result of these events. Given the human predictions and the algorithm predictions of those events, we asked an additional group to conclude which predictions seemed more logical. The results suggest that the ability of the algorithm to predict future events is at least as good as the human ability to predict.

To gain some intuition about the type of predictions the algorithm issues we present here two examples. The algorithm, given the event “Magnitude 6.5 earthquake rocks the Solomon Islands”, predicted that “Tsunami-warning will be issued in the Pacific Ocean”. It learnt this based on past examples it was trained on, one of which was “Tsunami warning issued for Indian-Ocean” after “7.6 earthquake strikes island near India”. The predictive template inferred by the algorithm was: if an earthquake occurs next to an Island, a tsunami warning will be issued for its nearest ocean. An additional example of a prediction issued by the algorithm, is given the event “Cocaine found at Kennedy Space Center”, it outputted the following predictions: “few people will be arrested”, as the past event “police found cocaine in lab” caused the event “2 people arrested”. As can be seen from the examples, not all of the predicted events will really take place in reality. However, most people will agree that they do seem logical.

The main contributions of this paper are threefold: First, we present a method of modeling events and construct the world’s largest causality graph, using novel causality mining techniques and data sources. Second, we present a novel method for prediction of general future events using their patterns in the past. Finally, we present a new architecture design for mining events, and a testing methodology for evaluating news prediction algorithms, and evaluate our algorithms.

2 Problem Definition

In this section we discuss the event-causality inference problem.

2.1 Event Representation

One of the theories to discuss how an event should be represented is Kim’s Property Exemplification of Events theory [Kim, 1993]. Kim’s discussion starts with the assumption of the availability of a set of entities O . These entities represent physical and abstract objects in the real world: people, instances, and types. Our representation of events complies with this theory, which, in

broad strokes, states that events are structured and defined by a triplet $[O, P, t]$: (1) An object or several objects ($O_i \subseteq O$); (2) a relation or property ($P \subseteq O \times O$), and (3) a time interval (t).

Kim’s theory provides a high-level view on events. To enhance the structure representation of an event we turn to Conceptual Dependency (CD) [Schank, 1972] theory. Inspired by this theory, we further structure the event to have roles in addition to the property relation. Each event will be composed of a temporal action or state that the event’s objects exhibit (P), the *actor* that conducted the action, the *object* on which the action was performed, the *instrument* the action was conducted with, and the *location* of the event. Formally, it is represented as an ordered set $e = \langle P, O_1, \dots, O_4, t \rangle$, where P is the action, $O_i \subseteq 2^O$ and t is a time-stamp. For example, the event “The U.S army destroyed a warehouse in Iraq with explosives”, which occurred on October 2004, is modeled as: Destroy (Action); U.S Army (Actor); warehouse (Object); explosives (Instrument); Iraq (Location); October 2004 (Time).

2.2 Learning Problem Definition

We treat causality inference as a learning problem. Let Ev be the set of events as described above. The goal concept is a set of ordered-pairs of events that share a causality relation among them: $Gc = \{\langle e_i, e_j \rangle | e_i, e_j \in Ev\}$, where e_i is an event causing e_j . Given training examples $T \subseteq Gc$, the algorithm produces a predictor $h : Ev \rightarrow Ev$ based on these examples. The predictor can then be applied on a new event $e_i \in Ev$ to output an element in the goal concept $\langle e_i, e_j \rangle \in Gc$.

2.3 Generalizations

The goal of learning is to generalize the training examples supplied, in order to be able to handle never seen new examples. The learning examples are composed of events, which in turn are composed of two main components – the objects and the properties that they hold at the time of the event. In this section we present how both components of the event can be generalized.

Generalizing Objects

A common practice is to map an object $o \in O$ to a concept $c \in C$. To allow generalization of the objects we assume the availability of a concept graph over those concepts. Formally, we assume a concept graph $G_O = (V, E)$, where $V \subseteq 2^O$. We also assume a labeling on the edges that represents the relation between the objects.

Generalizing Properties

In order to generalize the actions in the events we adopt the Conceptual Dependency (CD) paradigm of actions groups [Schank, 1972]. In this theory, Schank discusses how to map each action to 11 classes of actions, such as: move, PTrans, speak, Mbuild etc. This enables us to classify the event property P to higher action classes.

2.4 Hypothesis Space

We define an event $Gen(e) = e' = \langle P', O'_1, \dots, O'_4, t \rangle$ to be a generalization of an event $e = \langle P, O_1, \dots, O_4, t \rangle$, if there exists a path in the graph G_o between O_i to O'_i and if P' and P belong to the same class of actions. The hypothesis space is the set of all possible predictors $h : Ev \rightarrow Ev$. The hypothesis in our learning scheme is a generalized pair of events: $\langle Gen(e_i), Gen(e_j) \rangle$, $e_i \in Ev$.

3 The Prediction Algorithm

In this section we present a learning algorithm that learns how to generalize from past events in order to produce a predictor h , that given a present event can predict its effect. The solution builds on the case-based reasoning (CBR) framework [Aamodt and Plaza, 1994]. Given training examples and a new instance, this framework is described by 4 main stages: retrieving stored training examples most similar to the new instance (*retrieving*), combining the matched training instance with the new instance to produce a new solution (*reusing*), updating the stored instances (*revising*), and finally adding the new instance to the stored examples (*retaining*). In this work we implement the first two stages.

Retrieving Stage In order to define the similarity of two events, we first define the *Events Edit Distance* d_e . Let $e_i = \langle P^i, O_1^i, \dots, O_4^i, t^i \rangle$ and $e_j = \langle P^j, O_1^j, \dots, O_4^j, t^j \rangle$ be two events. Let G_o^u and G_p^u be the undirected versions of the objects graph G_o and the action graph G_p respectively¹. We define $d_e(e_i, e_j) = dist_{G_p^u}(P^i, P^j) + \sum_k dist_{G_o^u}(O_k^i, O_k^j)$, where, $dist_G$ is the length of the shortest path among all paths in G .

Given a new event e_i , the algorithm retrieves the top most similar events $S = \{\langle e, e_m \rangle \in T\}$ by d_e . Those matched events are then generalized: $Gen(S) = \{\langle e_1, e_2 \rangle | \exists e, e_m \in Ev : \langle e, e_m \rangle \in S, e_1 \in Gen(e), e_2 \in Gen(e_m)\}$. We define *specificity weight* of an event to be $\xi(e) = |\{\langle e, e_j \rangle \in S \cup Gen(S) | e, e_j \in Ev\}|$, representing the variance in the effects of the event e , and a *support score* $\vartheta(\langle e'_i, e'_j \rangle) = |\{\langle e_i, e_j \rangle \in S | e'_i \in Gen(e_i), e'_j \in Gen(e_j)\}|$, which indicates how many supportive evidence in the examples we have for this event pair or generalized event pair. The final similarity measurement of two events is: $Sim(e_i, e_j) = \frac{max_{e_m, \langle e_i, e_m \rangle \in Gen(S)} \vartheta(e_i, e_m) \cdot d_e(e_i, e_j)}{\xi(e_i)}$. In principal, we prefer matched events or generalized events which had many examples in the training (support) and cause little specific events (i.e. were not generalized too much). We apply the described distance on the cause events in $S \cup Gen(S)$, retrieving the top most similar events.

Reusing Stage Predicting the effects of the matched events directly has some drawbacks. Assume an event $e_i = \text{"Earthquake hits Haiti"}$ occurred today, and during retrieving, it was matched to the pair $\langle \text{"Earthquake hits$

Turkey" , "Red Cross help sent to Ankara" $\rangle$. Obviously, predicting that Red Cross help will be sent to Ankara because of an earthquake in Haiti is not logical. We would like to be able to generalize the past cause and effect pair and learn a predicate clause that connects them, e.g. for "Earthquake hits [Country Name]" yield "Red Cross help sent to [Capital of Country]". During the reusing stage, such a clause will be applied to the present event e_i producing its effect with regard to e_i . In our example, the logical predicate clause would be *CapitalOf*, as *CapitalOf(Turkey) = Ankara*. When applied on the current event e_i : *CapitalOf(Haiti) = Port-au-Prince*, the output will now be "Red Cross help sent to Port-au-Prince". Notice that the application of the clauses can only be applied on certain types of objects – in our case, countries. The clauses can be of any length, e.g., $\langle \text{"suspect arrested in Brooklyn"}, \text{"Bloomberg declares emergency"} \rangle$ produces the clause *Mayor(BoroughOf(x))*, as Brooklyn is a borough of New York, whose mayor is Bloomberg.

We will now show how to learn such clauses, and how they should be applied. Recall that the graph G_O is an edge-labeled graph, where each edge is a triplet $\langle v_1, v_2, l \rangle$, where l is a predicate (e.g. "CapitalOf"). The learning procedure is divided to 3 main steps: First, finding an undirected path p_i of length at most k in G_O between the objects of the cause event to the effect event; Second, constructing a clause using the labels of the path p_i as the predicates. We call this the *predicate projection* of size k , $pred = l_1, \dots, l_k$ from an event e_i to an event e_j . Finally, the projection is applied to the new event $e = \langle P^i, O_1, \dots, O_4, t \rangle$ by finding an undirected path in G_O from O_i with the labels of $pred$. The projection results are all the objects in the vertex reached. Formally, $pred$ can be applied if $\exists V_0 : O \subseteq V_0, \exists V_1 \dots V_k : (V_0, V_1, l_1), \dots, (V_{k-1}, V_k, l_k) \in Edges(G_O^u)$. The projection results are all the objects $o \in V_k$.

4 Mining Causality

In the previous section we present a high-level algorithm that requires training examples T , knowledge about entities G_O , and events' action classes P . One of the main challenges of this work was to build a scalable system to obtain those requirements.

We present a system that mines news sources to extract events, constructs their canonical semantic model, and builds a causality graph on top of those events. The system crawled, for more than 4 months, several dynamic information sources, the main one being the New-York-Times archives (which on part of the data optical character recognition (OCR) was performed), gathering data of more than 150 years (1851 – 2009).

For generalization of the objects, the system automatically reads web content and extracts world knowledge. The knowledge was mined from structured and semi-structured publicly available information repository. The causality graph building was distributed over 20 machines, using a Map-Reduce framework. This process efficiently unites different sources, extracts events,

¹The actions are grouped into classes, but we treat it as a 2-level graph.

and disambiguates entities. The resulting causality graph is composed of over 300 million entity nodes, one billion static edges and over 7 million causality edges.

On top of the causality graph, a search and indexing infrastructure was built to enable search over millions of documents. This highly scalable index allows a fast walk on the graph of events, enabling efficient inference capabilities during the reusing phase of the algorithm.

4.1 World knowledge mining

The entity graph G_o is composed of concepts from Wikipedia, ConceptNet[Liu and Singh, 2004], WordNet[Miller, 1995], Yago[Suchanek *et al.*, 2007], and OpenCyc. The concepts are interlinked using Linked-Data cloud (e.g., DBpedia). The billion labeled edges of the graph G_o are the predicates of those ontologies.

4.2 Causality events mining and extraction

Our supervised learning algorithm requires many learning examples to be able to generalize well. As the amount of temporal data is extremely large, spanning over millions of articles, the goal of getting human annotated examples becomes impossible. We therefore provide an automatic procedure to extract labeled examples for learning causality from dynamic content. Specifically in this work, we used the New-York-Times archives for the years 1851 – 2009, WikiNews and BBC – over 13 million articles in total.

The system mines unstructured natural language text, found in the dynamic web content, and searches for causal grammatical patterns. We construct those patterns using *causality connectors* [Wolff *et al.*, 2002; Levin and Hovav, 1994]. The connectors are divided to three groups: causal connectives (e.g. because, after), causal prepositions (e.g. due to, because of) and periphrastic causative verbs (e.g. cause, lead to). We constructed a set of rules for extracting a causality pair. Each rule is structured as: (Pattern, Constraint, Priority), where Pattern is a regular expression containing a causality connector, Constraint is a syntactic constraint on the sentence on which the pattern can be applied, and Priority is the priority of the rule if several rules can be matched. For example, for the causality connector “after”, the pattern “After [sentence1], [sentence2]” is used, with the constraint that [sentence1] cannot start with a number. This pattern can match the sentence “after Afghan vote, complaints of fraud surface” but will not match the sentence “after 10 years in Lansing, state lawmaker Tom George returns”. An additional pattern example is “[sentence1] as [sentence2]” with the constraint of [sentence2] having a verb. Using the constraint, the pattern can match the sentence “Nokia to cut jobs as it tries to catch up to rivals” is matched, but not the sentence “civil rights photographer unmasked as informer”. The result of a rule application is a pair of sentences – one tagged as a cause, and one tagged as an effect.

Given an extracted natural-language sentence, representing an event (either during learning or prediction),

the following procedure transforms it into a structured event:

1. Root forms of inflected words are extracted using a morphological analyzer derived from WordNet [Miller, 1995] stemmer. For example, in the article title from 10/02/2010: “U.S. attacks kill 17 militants in Pakistan”, the words “attacks”, “killed” and “militants” are transformed to “attack”, “kill” and “militant” respectively.
2. Part-Of-Speech tagging ([Marneffe *et al.*, 2006]) is performed, and the verb is identified. The class of the verb is identified using the VerbNet vocabulary [Hoa Trang Dang and Rosenzweig, 1998], e.g., kill belongs to $P = \text{murder}$ class.
3. A syntactic template matching the verb is applied to extract the semantic relations and thus the roles of the words. Those templates are based on VerbNet, which supplies for each verb class a set of syntactic templates. These templates match the syntax to the thematic roles of the entities in the sentence. We match the templates even if they are not continuous in the sentence tree. This allows the match of a sentence even where there is an auxiliary verb between the subject and the main transitive verb. In our example, the template is “NP1 V NP2” which transforms NP1 to “Agent”, and NP2 to “Patient”. Therefore, we match U.S. attacks to be the *Actor*, and the militant to be the *Patient*. If no template can be matched, the sentence is transformed into a typed-dependency graph of grammatical relations [Marneffe *et al.*, 2006]. In the example, U.S. attacks is identified as the subject of the sentence (candidate for Actor), militants as the object (candidate for Patient), and Pakistan as the preposition (candidate for Location or Instrument, based on heuristics, e.g., locations lexicons). Using this analysis, we identify that the *Location* is Pakistan.
4. Each word in O_i is mapped to a Wikipedia-based concept. If a word matches more than one concept, we perform disambiguation by computing the cosine similarity between the body of the news article and the body of the Wikipedia article associated with the concept: e.g., U.S was matched to several concepts, such as: United States, University of Salford, and Us (Brother Ali album). The most similar by content was United States Wikipedia concept.
5. The time of the event t is the time of the publication of the article in the news, e.g., $t = 10/02/2010$.

In our example, the final result is the event $e = \langle \text{Murder-Class, United States Of America, Militant, NULL, Pakistan, 10/02/2010} \rangle$.

In many cases additional heuristics were needed in order to deal with the briefness in news language, e.g:

1. Missing Context – In “McDonald’s recalls glasses due to Cadmium traces”, the extracted event “Cadmium traces” needs additional context – “Cadmium traces [in McDonald’s glasses]”. Heuristically, if an

object is missing, the first sentence ([sentence1]) subject is used.

2. Missing entities and verbs – the text “22 dead” should be structured to the event “22 [people] [are] dead”. Heuristically, if a number appears as the subject, the word people is added and used as the subject, and “be” is added as the verb.
3. Anaphora resolution – the text “boy hangs himself after he sees reports of Hussein’s execution” is modeled as “[boy₁] sees reports of Hussein’s execution” causes “[boy₁] hangs [boy₁]” [Lappin and Leass, 1994].
4. Negation – the text “Matsui is still playing despite his struggles” should be modeled as: “[Matsui] struggles” causes the event “Matsui is [not] playing”. Modeling preventive connectors (e.g., despite) requires negation of the modeled event.

5 Empirical Evaluation

A variety of experiments were conducted to test the performance and behavior of our algorithm.

5.1 Methodology

We implemented the algorithms described above and evaluated their performance. The prediction algorithm was trained using news articles from the period 1851 – 2009. The web resources snapshots mentioned in Section 4 dated until 2009. The evaluation is performed on separate data – Wikinews articles from the year 2010. We refer to this data as the *test data*. The evaluation procedure is divided to the following steps:

1. Event identification – our algorithm assumes that the input to the predictor h is an event. To find news titles that represent an event, we randomly sample n headlines from the test data. For each headline a human is requested to decide whether the headline is an event which can cause other events. We denote the set of headlines labeled as event as E . We again randomly sample k titles from E . We denote this group as C .
2. Algorithm event prediction – we run our algorithm on each event title $c_i \in C$. The algorithm performs event extraction from the headline, and produces an event e_i^a with the highest score of being caused by the event represented by c_i . The result of this stage are the pairs: $\{(c_i, e_i^a) | c_i \in C\}$.
3. Human event prediction – we present a human with an event title $c_i \in C$, asking what this event might cause. We denote the human result as e_i^h . The human is requested to provide the answer in a structured manner (as our algorithm produces). The result of this stage are the pairs: $\{(c_i, e_i^h) | c_i \in C\}$.
4. Human evaluation of the results – Present m people with a triplet (c_i, e_i^h, e_i^a) . We ask to evaluate the precision of the pair: (c_i, e_i^h) and the precision of (c_i, e_i^a) , on a scale of 0-4 (0 is a highly impossible prediction and 4 is a highly possible prediction).

	[0-1)	[1-2)	[2-3)	[3-4)	Average Ranking	Average Accuracy
Algorithm	0	2	19	29	3.08	77%
Humans	0	3	24	23	2.86	72%

Table 1: The histogram of the rankings of the users for both human and algorithm predictions.

Human evaluation was conducted using Amazon Mechanical Turk, an emerging utility for performing user study evaluations, which was shown to be very precise for certain tasks [Kittur *et al.*, 2008]. During the evaluation, tasks are created by routing a question to random users and obtaining their answers.

5.2 Results

We performed the above mentioned experiments, with the values $n = 1500, k = 50, m = 10$. The algorithm average prediction precision was 3.08/4 (3 is a “possible prediction”), and the human prediction average precision was 2.86/4. For each event, we average the results of the m rankers, producing an averaged score for the algorithm performance on the event, and an averaged score for the human prediction (see Table 1). We performed a paired t-test on the k paired scores. The advantage of the algorithm over the humans was found to be statistically significant with $p < 0.05$. We can conclude now that the ability of the algorithm to predict future events is at least as good as the human ability to predict.

We now present qualitative analysis of the results to have a better understanding of the algorithm strength and weaknesses. Given the event “Louisiana flood” the algorithm predicted that [number] people will flee. The prediction was based on the following past news articles: Residents of Florida flee storm and Hiltons; 150000 flee as hurricane nears north Carolina coast; a million flee as huge storm hits Texas coast; Thousands in Texas flee hurricane Ike; thousands flee as storm whips coast of Florida; at least 1000 flee flooding in Florida. The past events were generalized to the causality pair of “[Weather hazards] at [States of the Southern United States]” cause “[number] of people to flee”. During the prediction, the event “Louisiana flood” (which did not occur in the training examples) was found most similar to the above generalized causality pair.

As another example, given the event “6.1 magnitude aftershock earthquake hits Haiti”, it outputted the following predictions: “[number] people will be dead”, “[number] people will be missing”, “[number] magnitude aftershock earthquake will strike island near Haiti” and “earthquake will turn to United States Virgin Islands”. While the first 3 predictions seem very reasonable, the fourth one is problematic. The rule the system learnt in this case is – natural disasters that hit countries next to a shore tend to affect near by countries. In our case it predicted that the earthquake will affect United States Virgin Islands, which are geographically close to Haiti. However, the prediction “earthquake will turn to United States Virgin Islands” is not very realistic as an earth-

Event	Human-predicted event	Algorithm-predicted event
Al-Qaida demands hostage exchange	Al-Qaida exchanges hostage	A country will refuse the demand
Volcano erupts in Democratic Republic of Congo	Scientists in Republic of Congo investigate lava beds	Thousands of people flee from Congo
7.0 magnitude earthquake strikes Haitian coast	Tsunami in Haiti affects coast	Tsunami-warning is issued
2 Palestinians reportedly shot dead by Israeli troops	Israeli citizens protest against Palestinian leaders	War will be waged
Professor of Tehran University killed in bombing	Tehran students remember slain professor in memorial service	Professor funeral will be held
Alleged drug kingpin arrested in Mexico	Mafia kills people with guns in town	Kingpin will be sent to prison
UK bans Islamist group	Islamist group would adopt another name in the UK	Group will grow
China overtakes Germany as world's biggest exporter	German officials suspend tariffs	Wheat price will fall

Table 2: Human and algorithm predictions for events.

quake cannot change its course. It was created based on a match with a past example of a tornado hitting a country on a coast. The reason for that is the sparsity of the training. Both are natural disasters, and there were no negative examples or enough positive examples to support this distinction. However, we still find this example interesting, as it issues a prediction using spatial locality (United States Virgin Islands are [near] Haiti). Another example of the same problem is the prediction: { lightning kills 5 people, lightening will be arrested}, which was predicted based on training examples in which people who killed other people got arrested.

Additional 8 examples out of the 50 in the test and their predictions can be seen in Table 2.

6 Conclusions

We presented a method for representing events based on previous philosophical contribution. We introduced a novel case-based reasoning algorithm for predicting causality relations between events. Our prediction algorithm uses an extensive knowledge base which was automatically constructed using information mined from large amount of text. We presented the data mining and natural language techniques to transform the raw data of over 150 years of history archives into a structured representation of events, using a mined web-based object hierarchy and action classes. Our experimental evaluation showed that the predictions of the algorithm are at least as good as those of humans.

We believe that our work is one of the first to harness the vast amount of information available on the web to perform prediction that is general purpose, knowledge based, and human like.

References

- [Aamodt and Plaza, 1994] Agnar Aamodt and Enric Plaza. Case-based reasoning: foundational issues, methodological variations, and system approaches. *AI Commun.*, 7:39–59, 1994.

- [Carlson *et al.*, 2010] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E.R. Hruschka Jr., and T.M. Mitchell. Toward an architecture for never-ending language learning. In *Proc. AAAI*, 2010.
- [Chan and Lam, 2005] Ki Chan and Wai Lam. Extracting causation knowledge from natural language texts. *IJIC*, 20:327–358, 2005.
- [Hoa Trang Dang and Rosenzweig, 1998] Martha Palmer Hoa Trang Dang, Karin Kipper and Joseph Rosenzweig. Investigating regular sense extensions based on intersective levin classes. In *Proc. Coling-ACL*, 1998.
- [Kahneman and Tversky, 1973] D. Kahneman and A. Tversky. On the psychology of prediction. *Psychological review*, 80:237–251, 1973.
- [Kim, 1993] Jaegwan Kim. Supervenience and mind. *Selected Philosophical Essays*, 1993.
- [Kittur *et al.*, 2008] Aniket Kittur, H. Chi, and Bongwon Suh. Crowdsourcing user studies with mechanical turk. In *Proc. CHI*, 2008.
- [Lappin and Leass, 1994] Shalom Lappin and Herbert J. Leass. An algorithm for pronominal anaphora resolution. *Comput. Linguist.*, 20:535–561, 1994.
- [Levin and Hovav, 1994] B. Levin and M. Rappaport Hovav. A preliminary analysis of causative verbs in english. *Lingua*, 92:35–77, 1994.
- [Ling and Weld, 2010] Xiao Ling and Daniel S. Weld. Temporal information extraction. In *AAAI*, 2010.
- [Liu and Singh, 2004] H. Liu and P. Singh. Conceptnet: A practical commonsense reasoning toolkit. *BT Technology Journal*, 22, 2004.
- [Marneffe *et al.*, 2006] M. Marneffe, B. MacCartney, and C.D Manning. Generating typed dependency parses from phrase structure parses. In *Proc. LREC*, 2006.
- [Miller, 1995] George A. Miller. Wordnet: A lexical database for english. *CACM*, 38:39–41, 1995.
- [Schank, 1972] Roger Schank. Conceptual dependency: A theory of natural language understanding. *Cognitive Psychology* 3, 4:552–631, 1972.
- [Shahaf and Guestrin, 2010] Dafna Shahaf and Carlos Guestrin. Connecting the dots between news articles. In *Proc. KDD*, 2010.
- [Suchanek *et al.*, 2007] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proc. WWW*, 2007.
- [Wolff *et al.*, 2002] Phillip Wolff, Grace Song, and David Driscoll. Models of causation and causal verbs. In *Proc. ACL*, 2002.