PRISMATIC: Inducing Knowledge from a Large Scale Lexicalized Relation Resource*

James Fan and **David Ferrucci** and **David Gondek** and **Aditya Kalyanpur**
IBM Watson Research Lab
19 Skyline Dr
Hawthorne, NY 10532
{fanj, ferrucci, gondek, adityakal}@us.ibm.com

Abstract

One of the main bottlenecks in natural language processing is the lack of a comprehensive lexicalized relation resource that contains fine grained knowledge on predicates. In this paper, we present PRISMATIC, a large scale lexicalized relation resource that is automatically created over 30 gb of text. Specifically, we describe what kind of information is collected in PRISMATIC and how it compares with existing lexical resources. Our main focus has been on building the infrastructure and gathering the data. Although we are still in the early stages of applying PRISMATIC to a wide variety of applications, we believe the resource will be of tremendous value for AI researchers, and we discuss some of potential applications in this paper.

1 Introduction

Many natural language processing and understanding applications benefit from the interpretation of lexical relations in text (e.g. selectional preferences for verbs and nouns). For example, if one knows that things being annexed are typically geopolitical entities, then given the phrase *Napoleon's annexation of Piedmont*, we can infer *Piedmont* is a geopolitical entity. Existing linguistic resources such as VerbNet and FrameNet provide some argument type information for verbs and frames. However, since they are manually built, they tend to specify type constraints at a very high level (e.g, Solid, Animate), consequently they do not suffice for cases such as the previous example.

We would like to infer more fine grained knowledge for predicates automatically from a large amount of data. In addition, we do not want to restrict ourselves to only verbs, binary relations, or to a specific type hierarchy.

In this paper, we present PRISMATIC, a large scale lexicalized relation resource mined from over 30 gb of text. PRISMATIC is built using a suite of NLP tools that includes a dependency parser, a rule based named entity recognizer and a coreference resolution component. PRISMATIC is composed of frames which are the basic semantic representation of lexicalized relation and surrounding context. There are approximately 1 billion frames in our current version of PRISMATIC. To induce knowledge from PRISMATIC, we define the notion of frame-cuts, which basically specify a cut or slice operation on a frame. In the case of the previous Napoleon annexation example, we would use a noun-phrase → object type cut to learn the most frequent type of things being annexed. We believe there are many potential applications that can utilize PRISMATIC, such as type inference, relation extraction textual entailment, etc. We discuss some of these applications in details in section 8.

2 Related Work

2.1 Manually Created Resources

Several lexical resources have been built manually, most notably WordNet (Fellbaum, 1998), FrameNet(Baker et al., 1998) and VerbNet(Baker et

*Research supported in part by Air Force Contract FA8750-09-C-0172 under the DARPA Machine Reading Program

Proceedings of the NAACL HLT 2010 First International Workshop on Formalisms and Methodology for Learning by Reading, pages 122–127,
Los Angeles, California, June 2010.

al., 1998). WordNet is a lexical resource that contains individual word synset information, such as definition, synonyms, antonyms, etc. However, the amount of predicate knowledge in WordNet is limited.

FrameNet is a lexical database that describes the frame structure of selected words. Each frame represents a predicate (e.g. eat, remove) with a list of frame elements that constitutes the semantic arguments of the predicate. Different words may map to the same frame, and one word may map to multiple frames based on different word senses. Frame elements are often specific to a particular frame, and even if two frame elements with the same name, such as "Agent", may have subtle semantic meanings in different frames.

VerbNet is a lexical database that maps verbs to their corresponding Levin (Levin, 1993) classes, and it includes syntactic and semantic information of the verbs, such as the syntactic sequences of a frame (e.g. *NP V NP PP*) and the selectional restriction of a frame argument value *must be ANIMATE*,

Compared to these resources, in addition to being an automatic process, PRISMATIC has three major differences. First, unlike the descriptive knowledge in WordNet, VerbNet or FrameNet, PRISMATIC offers only numeric knowledge of the frequencies of how different predicates and their argument values through out a corpus. The statistical profiles are easily to produce automatically, and they allow additional knowledge, such as type restriction (see 8.1), to be inferred from PRISMATIC easily.

Second, the frames are defined differently. The frames in PRISMATIC are not abstract concepts generalized over a set of words. They are defined by the words in a sentence and the relations between them. Two frames with different slot values are considered different even though they may be semantically similar. For example, the two sentences "John loves Mary" and "John adores Mary" result in two different frame even though semantically they are very close. By choosing not to use frame concepts generalized over words, we avoid the problem of determining which frame a word belongs to when processing text automatically. We believe there will be enough redundancy in a large corpus to produce valid values for different synonyms and variations.

Third, PRISMATIC only uses a very small set of slots (see table 1) defined by parser and relation annotators to link a frame and its arguments. By using these slots directly, we avoid the problem of mapping parser relations to frame elements.

2.2 Automatically Created Resources

TextRunner (Banko et al., 2007) is an information extraction system which automatically extracts relation tuples over massive web data in an unsupervised manner. TextRunner contains over 800 million extractions (Lin et al., 2009) and has proven to be a useful resource in a number of important tasks in machine reading such as hypernym discovery (Alan Ritter and Etzioni, 2009), and scoring interesting assertions (Lin et al., 2009). TextRunner works by automatically identifying and extracting relationships using a conditional random field (CRF) model over natural language text. As this is a relatively inexpensive technique, it allows rapid application to web-scale data.

DIRT (Discovering Inference Rules from Text) (Lin and Pantel, 2001) automatically identifies inference rules over dependency paths which tend to link the same arguments. The technique consists of applying a dependency parser over 1 gb of text, collecting the paths between arguments and then calculating a path similarity between paths. DIRT has been used extensively in recognizing textual entailment (RTE).

PRISMATIC is similar to TextRunner and DIRT in that it may be applied automatically over massive corpora. At a representational level it differs from both TextRunner and DIRT by storing full frames from which n-ary relations may be indexed and queried. PRISMATIC differs from TextRunner as it applies a full dependency parser in order to identify dependency relationships between terms. In contrast to DIRT and TextRunner, PRISMATIC also performs co-reference resolution in order to increase coverage for sparsely-occurring entities and employs a named entity recognizer (NER) and relation extractor on all of its extractions to better represent intensional information.

3 System Overview

The PRISMATIC pipeline consists of three phases:

1. **Corpus Processing** Documents are annotated

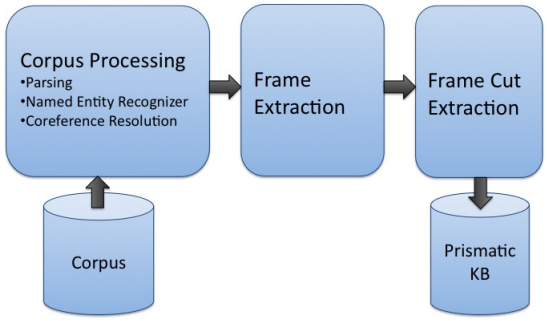

Figure 1: System Overview

by a suite of components which perform dependency parsing, co-reference resolution, named entity recognition and relation detection.

2. **Frame Extraction** Frames are extracted based on the dependency parses and associated annotations.

3. **Frame-Cut Extraction** Frame-cuts of interest (e.g. S-V-O cuts) are identified over all frames and frequency information for each cut is tabulated.

4 Corpus Processing

The key step in the Corpus Processing stage is the application of a dependency parser which is used to identify the frame slots (as listed in Table 1) for the Frame Extraction stage. We use ESG (McCord, 1990), a slot-grammar based parser in order to fill in the frame slots. Sentences frequently require co-reference in order to precisely identify the participating entity, and so in order to not lose that information, we apply a simple rule based co-reference resolution component in this phase. The co-reference information helps enhance the coverage of the frame-cuts, which is especially valuable in cases of sparse data and for use with complex frame-cuts.

A rule based Named Entity Recognizer (NER) is used to identify the types of arguments in all frame slot values. This type information is then registered in the Frame Extraction stage to construct intentional frames.

5 Frame Extraction

Relation	Description/Example
subj	subject
obj	direct object
iobj	indirect object
comp	complement
pred	predicate complement
objprep	object of the preposition
mod_nprep	*Bat Cave in Toronto is a tourist attraction.*
mod_vprep	*He made it to Broadway.*
mod_nobj	the object of a nominalized verb
mod_ndet	*City's budget was passed.*
mod_ncomp	*Tweet is a word for microblogging.*
mod_nsubj	*A poem by Byron*
mod_aobj	*John is similar to Steve.*
isa	subsumption relation
subtypeOf	subsumption relation

Table 1: Relations used in a frame and their descriptions

The next step of PRISMATIC is to extract a set of frames from the parsed corpus. A frame is the basic semantic unit representing a set of entities and their relations in a text snippet. A frame is made of a set of slot value pairs where the slots are dependency relations extracted from the parse and the values are the terms from the sentences or annotated types. Table 2 shows the extracted frame based on the parse tree in figure 2.

In order to capture the relationship we are interested in, frame elements are limited to those that represent the participant information of a predicate. Slots consist of the ones listed in table 1. Furthermore, each frame is restricted to be two levels deep at the most, therefore, a large parse tree may result in multiple frames. Table 2 shows how two frames are extracted from the complex parse tree in figure 2. The depth restriction is needed for two reasons. First, despite the best efforts from parser researchers, no parser is perfect, and big complex parse trees tend to have more wrong parses. By limiting a frame to be only a small subset of a complex parse tree, we reduce the chance of error parse in each frame. Second, by isolating a subtree, each frame focuses on the immediate participants of a predicate.

Non-parser information may also be included in a frame. For example, the type annotations of a word from a named entity recognizer are included, and such type information is useful for the various ap-

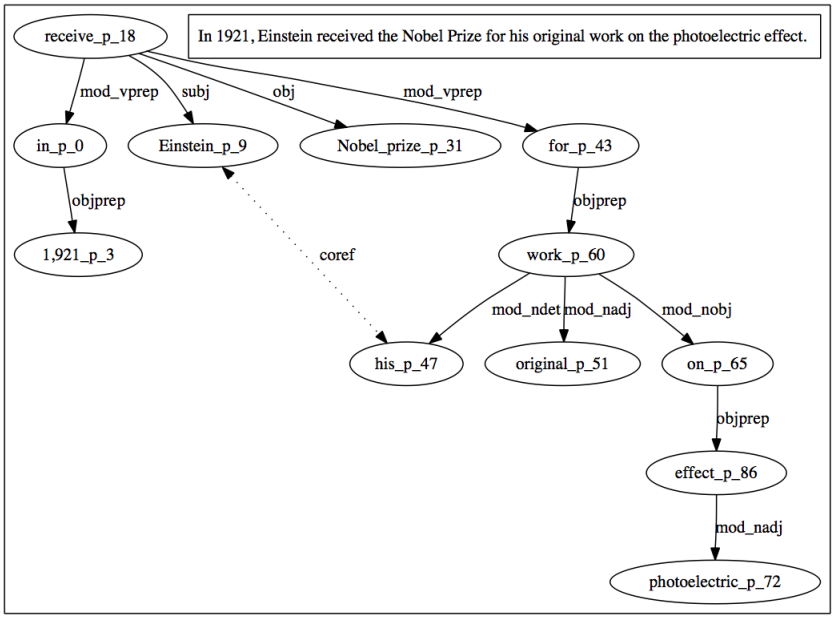

Figure 2: The parse tree of the sentence *In 1921, Einstein received the Nobel Prize for his original work on the photoelectric effect.*

Frame01	
verb	receive
subj	Einstein
type	PERSON / SCIENTIST
obj	Nobel prize
mod_vprep	in
objprep	1921
type	YEAR
mod_vprep	for
objprep	Frame02

Frame02	
noun	work
mod_ndet	his / Einstein
mod_nobj	on
objprep	effect

Table 2: Frames extracted from Dependency Parse in Figure 2

plications described in section 8. We also include a flag to indicate whether a word is proper noun. These two kinds of information allow us to easily separate the intensional and the extensional parts of PRISMATIC.

6 Frame Cut

One of the main reasons for extracting a large amount of frame data from a corpus is to induce interesting knowledge patterns by exploiting redundancy in the data. For example, we would like to learn that things that are annexed are typically regions, i.e., a predominant object-type for the noun-phrase "annexation of" is "Region" where "Region" is annotated by a NER. To do this kind of knowledge induction, we first need to abstract out specific portions of the frame - in this particular case, we need to isolate and analyze the noun-phrase → object-type relationship. Then, given a lot of data, and frames containing only the above relationship, we hope to see the frame [noun="annexation", preposition="of", object-type="Region"] occur very frequently.

To enable this induction analysis, we define frame-cuts, which basically specify a cut or slice operation on a frame. For example, we define an N-P-OT frame cut, which when applied to a frame only keeps the noun (N), preposition (P) and object-type (OT) slots, and discards the rest. Similarly, we define frame-cuts such as S-V-O, S-V-O-IO, S-V-P-O etc. (where S - subject, V - verb, O - object, IO - indirect object) which all dissect frames along dif-

ferent dimensions. Continuing with the annexation example, we can use the V-OT frame cut to learn that a predominant object-type for the verb "annex" is also "Region", by seeing lots of frames of the form [verb="annex", object-type="Region"] in our data.

To make frame-cuts more flexible, we allow them to specify optional value constraints for slots. For example, we can define an S-V-O frame cut, where both the subject (S) and object (O) slot values are constrained to be proper nouns, thereby creating strictly extensional frames, i.e. frames containing data about instances, e.g., [subject="United States" verb="annex" object="Texas"]. The opposite effect is achieved by constraining S and O slot values to common nouns, creating intensional frames such as [subject="Political-Entity" verb="annex" object="Region"]. The separation of extensional from intensional frame information is desirable, both from a knowledge understanding and an applications perspective, e.g. the former can be used to provide factual evidence in tasks such as question answering, while the latter can be used to learn entailment rules as seen in the annexation case.

7 Data

The corpora we used to produce the initial PRISMATIC are based on a selected set of sources, such as the complete Wikipedia, New York Times archive and web page snippets that are on the topics listed in wikipedia. After cleaning and html detagging, there are a total of 30 GB of text. From these sources, we extracted approximately 1 billion frames, and from these frames, we produce the most commonly used cuts such as S-V-O, S-V-P-O and S-V-O-IO.

8 Potential Applications

8.1 Type Inference and Its Related Uses

As noted in Section 6, we use frame-cuts to dissect frames along different slot dimensions, and then aggregate statistics for the resultant frames across the entire dataset, in order to induce relationships among the various frame slots, e.g., learn the predominant types for subject/object slots in verb and noun phrases. Given a new piece of text, we can apply this knowledge to infer types for named entities. For example, since the aggregate statistics shows the most common type for the object of the verb "annex" is Region, we can infer from the sentence "Napoleon annexed Piedmont in 1859", that "Piedmont" is most likely to be a Region. Similarly, consider the sentence: "He ordered a Napoleon at the restaurant". A dictionary based NER is very likely to label "Napoleon" as a Person. However, we can learn from a large amount of data, that in the frame: [subject_type="Person" verb="order" object_type=[?] verb_prep="at" object_prep="restaurant"], the object_type typically denotes a Dish, and thus correctly infer the type for "Napoleon" in this context. Learning this kind of fine-grained type information for a particular context is not possible using traditional hand-crafted resources like VerbNet or FrameNet. Unlike previous work in selectional restriction (Carroll and McCarthy, 2000; Resnik, 1993), PRISMATIC based type inference does not dependent on a particular taxonomy or previously annotated training data: it works with any NER and its type system.

The automatically induced-type information can also be used for co-reference resolution. For example, given the sentence: "Netherlands was ruled by the UTP party before Napolean annexed it", we can use the inferred type constraint on "it" (Region) to resolve it to "Netherlands" (instead of the "UTP Party").

Finally, typing knowledge can be used for word sense disambiguation. In the sentence, "Tom Cruise is one of the biggest stars in American Cinema", we can infer using our frame induced type knowledge base, that the word "stars" in this context refers to a Person/Actor type, and not the sense of "star" as an astronomical object.

8.2 Factual Evidence

Frame data, especially extensional data involving named entities, captured over a large corpus can be used as factual evidence in tasks such as question answering.

8.3 Relation Extraction

Traditional relation extraction approach (Zelenko et al., 2003; Bunescu and Mooney, 2005) relies on the correct identification of the types of the argument. For example, to identify "employs" relation between "John Doe" and "XYZ Corporation", a relation extractor heavily relies on "John Doe" being annotated

as a "PERSON" and "XYZ Corporation" an "ORGANIZATION" since the "employs" relation is defined between a "PERSON" and an "ORGANIZATION".

We envision PRISMATIC to be applied to relation extraction in two ways. First, as described in section 8.1, PRISMATIC can complement a named entity recognizer (NER) for type annotation. This is especially useful for the cases when NER fails. Second, since PRISMATIC has broad coverage of named entities, it can be used as a database to check to see if the given argument exist in related frame. For example, in order to determine if "employs" relation exists between "Jack Welch" and "GE" in a sentence, we can look up the SVO cut of PRISMATIC to see if we have any frame that has "Jack Welch" as the subject, "GE" as the object and "work" as the verb, or frame that has "Jack Welch" as the object, "GE" as the subject and "employs" as the verb. This information can be passed on as an feature along with other syntactic and semantic features to th relation extractor.

9 Conclusion and Future Work

In this paper, we presented PRISMATIC, a large scale lexicalized relation resource that is built automatically over massive amount of text. It provides users with knowledge about predicates and their arguments. We have focused on building the infrastructure and gathering the data. Although we are still in the early stages of applying PRISMATIC, we believe it will be useful for a wide variety of AI applications as discussed in section 8, and will pursue them in the near future.

References

Stephen Soderland Alan Ritter and Oren Etzioni. 2009. What is this, anyway: Automatic hypernym discovery. In *Proceedings of the 2009 AAAI Spring Symposium on Learning by Reading and Learning to Read*.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The berkeley framenet project. In *Proceedings of the 17th international conference on Computational linguistics*, pages 86–90, Morristown, NJ, USA. Association for Computational Linguistics.

Michele Banko, Michael J Cafarella, Stephen Soderl, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *In International Joint Conference on Artificial Intelligence*, pages 2670–2676.

Razvan C. Bunescu and Raymond J. Mooney. 2005. A shortest path dependency kernel for relation extraction. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 724–731, Morristown, NJ, USA. Association for Computational Linguistics.

John Carroll and Diana McCarthy. 2000. Word sense disambiguation using automatically acquired verbal preferences. *Computers and the Humanities Senseval Special Issue*, 34.

Christiane Fellbaum, 1998. *WordNet: An Electronic Lexical Database*.

Beth Levin, 1993. *English Verb Classes and Alternations: A Preliminary Investigation*.

Dekang Lin and Patrick Pantel. 2001. Dirt - discovery of inference rules from text. In *In Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 323–328.

Thomas Lin, Oren Etzioni, and James Fogarty. 2009. Identifying interesting assertions from the web. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 1787–1790, New York, NY, USA. ACM.

Michael C. McCord. 1990. Slot grammar: A system for simpler construction of practical natural language grammars. In *Proceedings of the International Symposium on Natural Language and Logic*, pages 118–145, London, UK. Springer-Verlag.

Philip Resnik. 1993. *Selection and Information: A Class-Based Approach to Lexical Relationships*. Ph.D. thesis, University of Pennsylvania.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *J. Mach. Learn. Res.*, 3:1083–1106.