Implementing Argumentation Schemes as Logic Programs

Nancy L. Green
University of North Carolina Greensboro, Greensboro, NC, USA
nlgreen@uncg.edu

Abstract

The dominant approach to argumentation mining has been to treat it as a text classification problem. However some applications to scientific text, such as accurately summarizing argumentation in research articles, require a deeper understanding of the text. This paper provides a novel approach in which argumentation schemes are represented as logic program rules for use in argumentation mining. The logic programs can be used, not only to recognize fully explicit arguments, but also arguments with implicit conclusions. This paper presents seven implemented rules based on analysis of an open-access biomedical research article. The rules are specializations of general schemes that can apply to other qualitative causal domains in the natural sciences.

1 Introduction

The dominant approach in the relatively new field of argumentation mining [e.g., Green et al., 2014; Cardie et al., 2015] has been to treat it as a machine learning problem, enabling researchers to adopt methods that have been applied successfully to other natural language processing tasks such as sentiment analysis and information extraction. That general approach can be useful for certain types of applications such as classification of sentiment as positive or negative in social media, or classification of sentences as premise or conclusion for the purpose of automatic assessment of student essay quality. However some applications, such as accurately summarizing argumentation in scientific research articles, require a deeper understanding of the text.

There are a number of problems with mining argumentation at the text level (sentence, clause, or smaller phrases) rather than at the semantic level [Green, 2015a; 2015b]. Often scientific text contains enthymemes, i.e. arguments with implicit premises or an implicit conclusion. Interpretation of enthymemes may require use of the preceding discourse context (including inferred conclusions of other arguments), presumed shared knowledge of the author and audience, as well as constraints of the underlying argumentation scheme [Green, 2010]. Furthermore, explicitly given components may not occur in contiguity with other components of the same argument. In fact, the content of two arguments may be interleaved at the text level.

Although human-level understanding of natural language text is currently beyond the state of the art, we contend that an inference-based approach is feasible for scientific applications requiring a deeper analysis of argumentation. In support of this position, this paper demonstrates a novel approach in which argumentation schemes are implemented as logic programs. In addition, this paper explains how this inference-based approach fits into an argumentation mining system architecture.

2 Representing Biomedical Arguments

Argumentation schemes are abstract descriptions of acceptable, possibly defeasible, arguments used in conversation as well as in formal genres such as legal and scientific text [Walton et al., 2008]. As a step towards argumentation mining scientific text, in previous work we described some argumentation schemes used in biomedical research articles on human genetic variants with adverse health effects [Green, 2015a; 2015b]. Designed for use by human analysts for creation of annotated corpora, the descriptions were given at a level of abstraction applicable to argumentation not just in biomedicine but also in other domains.

In order to enable computer programs to represent and reason about scientific argumentation, this paper shows how to represent key argumentation schemes as logic program rules written in Prolog [Bratko, 2001]. The schemes are 'key' in the sense that they are specializations of schemes that can apply to other qualitative causal domains in the natural sciences. In our approach, the argumentation scheme and its premises and conclusion are recognized at the same time. This is in contrast to surface text-based machine-learning approaches to argumentation scheme identification, e.g. [Feng and Hirst, 2011]. In that approach, clauses must be labeled as premise or conclusion before argumentation scheme recognition is performed. Another significant feature of our

approach is that the rules only make use of semantic predicates, rather than superficial text features, such as the presence of a certain discourse connective. As discussed in Section 3, the semantic predicates refer to a partial semantic interpretation of the text, as well as limited domain knowledge.

To provide argumentation schemes for freely available text to the argumentation mining research community in this paper, we analyzed argumentation schemes in the Results section of an open-access biomedical research article (CRAFT175900787) [van de Leemput et al., 2007] in the CRAFT corpus [CRAFT]. The CRAFT corpus has been annotated by other researchers for purposes of biomedical text mining [Verspoor et al., 2012; Bada et al., 2012], but not for argumentation mining. Based upon analysis of the CRAFT article we implemented the seven causal argumentation schemes shown in Figures 1 to 7. Some of the schemes were used in more than one argument in the article. Note that the conclusions of these schemes are not asserted with complete certainty. The corresponding arguments in the source text range in force from 'plausible hypothesis' to 'fairly certain conclusion'. It is outside of the scope of this paper to address modality, although this is an important issue for our future research.

Figure 1 shows the logic program for recognizing an argument following a pattern similar to Mill's Method of Agreement [Jenicek and Hitchcock, 2005]. Note that in this and in the other logic programs defining the argumentation schemes given in Figures 1 to 7, domain-specific predicates are used. However, these argumentation schemes can be seen as specializations of more general descriptions, e.g. as given in [Green, 2015a; 2015b] or [Walton et al., 2008]. The source of domain knowledge used in the rules is covered in Section 3.

As exemplified in Figure 1, the seven schemes have been implemented in such a way that not only the argumentation schemes but also their conclusions can be inferred by the rule. The motivation for so implementing the schemes is that in this genre the text often contains enthymemes, arguments with implicit premises or an implicit conclusion. After an implicit conclusion has been inferred it can be added to the knowledge base used in recognition of argumentation schemes (Section 3). Then it can be used to recognize a subsequent argument it which it functions as an implicit premise.

The scheme shown in Figure 2 differs from the preceding scheme as follows: the premise in Figure 1 referring to the expected phenotype P is negated and the conclusion in Figure 1 that genotype M causes P is negated. Note that the implementation in Figure 2 and several other of the schemes in this paper makes use of a 'knot' operator that we defined to mean 'known not', to be distinguished from Prolog's negation operator 'not', which means 'cannot be proven'.

```
arg(
 scheme('Agreement'),
 premise(have_phenotype(G, P)),
 premise(have_genotype(G, M)),
 conclusion(cause(M, P)))
:-
group(G),
have_phenotype(G, P),
have_genotype(G, M).
```

Paraphrase: For all G, P, M
Premises:

- Group G has phenotype P
- Group G has genotype M

Conclusion: M causes P.

Figure 1. Method of Agreement

```
arg(
 scheme('Failed-Agreement-effect'),
 premise(knot(have_phenotype(G, P))),
 premise(have_genotype(G, M)),
 conclusion(knot(cause(M, P))))
:-
group(G),
knot(have_phenotype(G, P)),
have_genotype(G, M).
```

Paraphrase: For all G, P, M
Premises:

- Group G does not have phenotype P
- Group G has genotype M

Conclusion: M does not cause P.

Figure 2. Method of Failed Agreement (no effect)

The scheme shown in Figure 3 is a specialization of Mill's Method of Difference [Jenicek and Hitchcock, 2005]. The scheme shown in Figure 4 is a specialization of Argument by Analogy described by argumentation theorists, e.g. [Walton et al., 2008]. An example is given in the Appendix.

```
arg(
 scheme('Difference'),
 premise(have_phenotype(G1, P1)),
 premise(have_genotype(G1, M1)),
 premise(knot(have_phenotype(G2, P1))),
 premise(knot(have_genotype(G2, M1))),
 conclusion(cause(M1, P1)))
:-
group(G1), group(G2), not(G1=G2),
have_phenotype(G1,P1),
have_genotype(G1,M1),
knot(have_phenotype(G2,P1)),
knot(have_genotype(G2, M1)).
```

Paraphrase: For all G1, G2, P1, P2, M
Premises:

- Group G1 has phenotype P
- Group G1 has genotype M
- Group G2 does not have phenotype P
- Group G2 does not have genotype M

Conclusion: M causes P.

Figure 3. Method of Difference

```
arg(
 scheme('Analogy'),
 premise(have_phenotype(G1, P1)),
 premise(have_phenotype(G2, P2)),
 premise(similar(P1, P2)),
 premise(have_genotype(G1, M1)),
 premise(have_genotype(G2, M2)),
 premise(similar(M1, M2)),
 premise(cause(M1, P1)),
 conclusion(cause(M2, P2)))
:-
group(G1), group(G2), not(G1=G2),
have_phenotype(G1, P1),
have_phenotype(G2, P2),
have_genotype(G1, M1),
have_genotype(G2, M2),
cause(M1, P1),
similar(P1, P2), similar(M1, M2).
```

Paraphrase: For all G1, G2, P1, P2, M1, M2
Premises:

- Group G1 has phenotype P1
- Group G2 has phenotype P2, where P2 is similar to P1
- Group G1 has genotype M1
- Group G2 has genotype M2, where M2 is similar to M1
- M1 causes P1

Conclusion: M2 causes P2.

Figure 4. Analogy

The scheme shown in Figure 5 is very similar to Method of Difference (Figure 3). In Method of Difference, the presence/absence of a potential causal agent is correlated with the presence/absence of an observed potential effect. In Eliminate Difference, the presence/absence of the potential causal agent A is implied, based upon the knowledge that A is the difference in the presence of AB and of B.

```
arg(
 scheme('Eliminate Difference'),
 premise(have_phenotype(G1, P)),
 premise(have_genotype(G1, AB)),
 premise(knot(have_phenotype(G2, P))),
 premise(have_genotype(G2, B)),
 conclusion(cause(A, P)))
:-
group(G1), group(G2), not(G1=G2),
difference(AB,B,A),
have_phenotype(G1,P),
have_genotype(G1,AB),
knot(have_phenotype(G2,P)),
have_genotype(G2,B).
```

Paraphrase: For all G1, G2, P, AB, B, A
Premises:

- Group G1 has phenotype P
- Group G2 does not have phenotype P
- Group G1 has genotype AB
- Group G2 has genotype B
- Genotype A is the difference between AB and B.

Conclusion: A causes P.

Figure 5. Eliminate Difference.

The preceding argumentation schemes can be challenged by means of at least two *critical questions*:

- Is there an alternative causal agent?
- Is there a plausible causal mechanism explaining how the putative causal agent can lead to the observed effect?

Arguments constructed from the next two schemes are used to respond to the second of those questions.

The scheme shown in Figure 6 describes a causal chain that explains how in one group (G1) a genotype (M1) caused a certain phenotype (P1) by means of a protein abnormality (Prot) caused by M1 and which is associated with P1. Then, as the argument goes, since in another group (G2) its genotype (M2) is similar to M1, and its phenotype (P2) is similar to P1, and M2 causes the same protein abnormality (Prot) in

G2, and Prot is also associated with P2 in G2, then (it is plausible that) M2 caused P2.

```
arg(
 scheme('Consistent Explanation'),
 premise(have_genotype(G1, M1)),
 premise(have_protein(G1, Prot)),
 premise(have_phenotype(G1, P1)),
 premise(cause(M1, Prot)),
 premise(assoc(Prot, P1)),
 premise(cause(M1, P1)),
 premise(have_genotype(G2, M2)),
 premise(have_protein(G2, Prot)),
 premise(have_phenotype(G2, P2)),
 premise(similar(M1, M2)),
 premise(similar(P1, P2)),
 premise(cause(M2, Prot)),
 premise(assoc(Prot, P2)),
 conclusion(cause(M2, P2)))
:-
group(G1),  group(G2), not(G1=G2),
similar(M1, M2), similar(P1, P2),
have_genotype(G1, M1),
have_protein(G1, Prot),
have_phenotype(G1, P1),
cause(M1, Prot),
assoc(Prot, P1),
cause(M1, P1),
have_genotype(G2, M2),
have_protein(G2, Prot),
have_phenotype(G2, P2),
cause(M2, Prot),
assoc(Prot, P2).
```

Paraphrase: For all G1, G2, M1, M2, Protein, P1, P2
Premises:

- Group G1 has genotype M1
- Group G1 has protein Prot
- Group G1 has phenotype P1
- M1 causes Prot
- Prot is associated with P1
- M1 causes P1
- Group G2 has genotype M2, where M2 is similar to M1
- Group G2 has protein Prot
- Group G2 has phenotype P2, where P2 is similar to P1
- M2 causes Prot
- Prot is associated with P2

Conclusion: M2 causes P2.

Figure 6. Consistent Explanation.

The scheme shown in Figure 7 combine aspects of Method of Difference (Figure 3) and Consistent Explanation (Figure 6).

```
arg(
 scheme('Difference Consistent Explanation'),
 premise(have_genotype(G1, M)),
 premise(have_protein(G1, Prot)),
 premise(have_phenotype(G1, P)),
 premise(cause(M1, Prot)),
 premise(assoc(Prot, P)),
 premise(knot(have_genotype(G2, M))),
 premise(knot(have_protein(G2, Prot))),
 premise(knot(have_phenotype(G2, P))),
 conclusion(cause(M, P)))
:-
group(G1), group(G2), not(G1=G2),
have_genotype(G1, M),
have_protein(G1, Prot),
have_phenotype(G1, P),
cause(M1, Prot),
assoc(Prot, P),
knot(have_genotype(G2, M)),
knot(have_protein(G2, Prot)),
knot(have_phenotype(G2, P)).
```

Paraphrase: For all G1, G2, M, Protein, P
Premises:

- Group G1 has genotype M
- Group G1 has protein Prot
- Group G1 has phenotype P
- M causes Prot
- Prot is associated with P
- Group G2 does not have phenotype P
- Group G2 does not have genotype M
- Group G2 does not have protein Prot

Conclusion: M causes P

Figure 7. Difference Consistent Explanation

The above seven argumentation schemes were not the only schemes that we found, but are presented here as they seem to be the most useful to researchers working in other domains in the natural sciences. The implemented rules have been tested using a manually created knowledge base. The next section describes how such a knowledge base would be created in an argumentation mining system.

3 System Architecture

It is assumed that, before argumentation scheme recognition using the rules given in the previous section begins, current biomedical/biological natural language processing tools would be applied to a source text to create a knowledge base. Named entity recognition tools such as ABNER (Settles 2005) or MutationFinder (Caporaso et al. 2007) could be used to recognize expressions referring to semantic class names such as genes, mutations, proteins, and phenotypes. Domain-specific relations in the rules, *have_phenotype*, *have_genotype*, and *have_protein*, could be extracted from the text using relation extraction tools such as OpenMutationMinder (Naderi and Witte 2012). Relations *cause* and *assoc* (association) could be extracted by domain-specific tools as well as non-domain-specific discourse coherence relation extraction tools. Also, a certain amount of domain knowledge would be required, i.e., for the relations *similar* and *difference*, which could be acquired from a domain ontology or domain experts.

For an example showing an excerpt from the CRAFT article, a representation of the knowledge needed to recognize the Argument by Analogy given in that paragraph, and the argument that is recognized, see the Appendix.

4 Discussion

Previous argumentation mining research has not addressed the natural sciences. However, argumentation is an important feature of scientific discourse. Previous investigations of scientific discourse addressed automatic classification of text segments without requiring semantic interpretation of a text, e.g., classifying segments' discourse coherence relations [Prasad et al., 2011], epistemological status (hypothesis, background knowledge, new knowledge claim, etc.) [Teufel, 2010], or component of a scientific investigation (hypothesis, method, result, etc.) [Liakata, 2012].

In contrast, this paper demonstrated a semantic approach to automatic recognition of premises, conclusion, and argumentation scheme of arguments in scientific text. In this approach, argumentation schemes are implemented as logic programs. The logic programs would be used with a knowledge base that could be constructed from a text in a large part automatically using existing language processing tools. The logic programs can be used, not only to recognize fully explicit arguments in the text, but also arguments with implicit conclusions. This is important because often the conclusions are implicit and may function as implicit premises of subsequent arguments in the text. Although the argumentation schemes presented here have been implemented using domain-specific predicates, they are specializations of more general schemes applicable to other qualitative causal domains in the natural sciences. Thus we expect that researchers can adapt them to other domains and begin argumentation mining in those domains using a similar approach.

References

[Bada et al., 2012] M. Bada, M. Eckert, D. Evans, et al. Concept Annotation in the CRAFT corpus. *BMC Bioinformatics*, 13:161, 2012.

[Bratko, 2001] Ivan Bratko. *Prolog Programming for Artificial Intelligence*. 3rd edition. Addison-Wesley, Harlow, England, 2001.

[Caporaso et al. 2007] J. G. Caporaso, W.A. Baumgartner Jr., D. A. Randolph, K. B. Cohen, and L. Hunter. MutationFinder: A high-performance system for extracting point mutation mentions from text. *Bioinformatics*, 23:1862-1865.

[Cardie et al., 2015] C. Cardie et al. (Eds.) *Second Workshop on Argumentation Mining*. North American Conference of the Association for Computational Linguistics, Denver, 2015.

[CRAFT] http://bionlp-corpora.sourceforge.net /CRAFT/ index.shtml.

[Feng and Hirst, 2011] V.W. Feng and G. Hirst. Classifying Arguments by Scheme. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics,* pages 987-996, Portland, OR, 2011.

[Green, 2010] N. Green. Representation of Argumentation in Text with Rhetorical Structure Theory. *Argumentation* 24(2): 181-196.

[Green, 2015a] N. Green. Identifying Argumentation Schemes in Genetics Research Articles. In *Proceedings of the Second Workshop on Argumentation Mining*, North American Conference of the Association for Computational Linguistics (NAACL), Denver, CO, 2015.

[Green, 2015b] N. Green. Annotating Evidence-Based Argumentation in Biomedical Text. In *Proc. 2015 Int. Workshop on Biomedical and Health Informatics, IEEE Int. Conf. on Bioinformatics and Biomedicine (BIBM 2015), Washington, D.C, Nov. 9-12, 2015*. IEEE Computer Society Press.

[Green et al., 2014] N. Green, et al. (Eds.) *First Workshop on Argumentation Mining.* Association for Computational Linguistics, Baltimore, MD., 2014.

[Jenicek and Hitchcock, 2005] M. Jenicek and D. Hitchcock. *Logic and Critical Thinking in Medicine*. American Medical Association Press, 2005.

[van de Leemput et al., 2007] J. van de Leemput, J. Chandran, M. Knight, et al. Deletion at *ITPR1* Underlies Ataxia in Mice and Spinocerebellar Ataxia 15 in Humans. *PLoS Genetics*, 3(6) e108:1076-1082, 2007.

[Liakata, M. et al., 2012] M. Liakata. Automatic recognition of conceptualization zones in scientific articles and two life science applications. *Bioinformatics* 28(7), 2012.

[Naderi and Witte, 2012] N. Naderi and R. Witte. Automated extraction and semantic analysis of mutation impacts from the biomedical literature. *BMC Genomics*, 13(Suppl 4):510, 2012.

[Prasad et al., 2011] R.Prasad, S. McRoy, N. Frid, A. Joshi, and H. Yu. The Biomedical Discourse Relation Bank. *BMC Bioinformatics*, 12:188, 2011.

[Settles, 2005] B. Settles, B. ABNER: an open source tool for automatically tagging genes, proteins, and other entity names in text. *Bioinformatics* 21(14):3191-3192, 2005.

[Teufel, 2010] S. Teufel, S. *The Structure of Scientific Articles: Applications to Citation Indexing and Summarization*. Stanford, CA, CSLI Publications.

[Verspoor et al., 2012] K. Verspoor, K.B. Cohen, A. Lanfranchi, et al. A Corpus of Full-text Journal Articles is a Robust Evaluation Tool for Revealing Differences in Performance of Biomedical Natural Language Processing Tools. *BMC Bioinformatics* 13:207, 2012.

[Walton et al., 2008] D. Walton, C. Reed, and F. Macagno. *Argumentation Schemes.* Cambridge University Press, 2008.

Appendix.

Excerpt from [van de Leemput et al., 2007] (CRAFT175900787):

Literature searches revealed that among disease lines mapped to 6qE1, the spontaneous mutant opt mouse displays a strikingly similar presentation to that described here [1]. The underlying genetic lesion causing the opt phenotype is a homozygous in-frame deletion of exons 43 and 44 of the gene Itpr1 (Itpr1opt/opt), encoding inositol 1,4,5-triphosphate receptor 1 (Itpr1). Sequencing of all exons and intron–exon boundaries of Itpr1 in affected mice from the current study revealed a single mutation within Itpr1: a novel in-frame deletion of 18 bp within exon 36 (Itpr1Δ18/Δ18).

Semantic relations extracted from excerpt:

group(opt).
have_phenotype(opt, opt_pheno).
have_genotype(opt, 'Itpr1opt/opt').
have_genotype(affected_knockout_mice,
' Itpr1Δ18/Δ18').
cause('Itpr1opt/opt', opt_pheno).
similar(opt_pheno, ataxia).

Semantic relations extracted from text preceding the excerpt:

group(affected_knockout_mice).
have_phenotype(affected_knockout_mice, ataxia).

Domain knowledge:

similar('Itpr1opt/opt', ' Itpr1Δ18/Δ18').

Recognized argument:

Scheme: Analogy
Premises:
have_phenotype(opt, opt_pheno).
have_genotype(opt, 'Itpr1opt/opt').
have_phenotype(affected_knockout_mice,
ataxia).
have_genotype(affected_knockout_mice,
' Itpr1Δ18/Δ18').
cause('Itpr1opt/opt', opt_pheno).
similar(opt_pheno, ataxia).
similar('Itpr1opt/opt', ' Itpr1Δ18/Δ18').
Conclusion:
cause(' Itpr1Δ18/Δ18', ataxia).