A First Experiment Using ILP for Argument Mining

Nancy L. Green[1] and L. Joshua Crotts[1]

[1] *University of North Carolina Greensboro, 1400 Spring Garden Street, Greensboro, NC 27402, USA*

Abstract

This paper presents a first experiment using Inductive Logic Programming (ILP) to acquire argument schemes from a genetics research paper in which entities, relations, and arguments had been annotated previously. Then the ILP-derived rules are used to extract argument premises and conclusions from the text. Another contribution of this paper is a computational model of the narrative of scientific discovery in the text.

Keywords

Argument Mining, Argument Schemes, Inductive Logic Programming, Science Discovery Narrative, Biological/Biomedical Sciences Research Literature

1. Introduction

In [5], we argued the need for a semantics-informed approach to argument mining genetics research papers and proposed how to do that as follows. First, BioNLP tools [15] would be used to identify entities and relations in the text. Next, argument schemes implemented as logic-programming rules in terms of those entities and relations could be used to extract individual arguments. To demonstrate that approach, we manually annotated entities, a small set of domain relations, and arguments in the Results/Discussion section of a genetics research paper [19]. Based on the annotations, we implemented seven argument schemes as Prolog rules. We suggested that in the future such rules might be acquired by application of Inductive Logic Programming (ILP) [13] to corpora in which entities, relations, and arguments had been annotated. ILP is a semantically rich machine learning technique which can exploit background knowledge expressed as logic programs. Another advantage of use of ILP is that only a small set of examples is needed, in contrast to other approaches to machine learning.

This paper presents a first experiment using ILP to acquire argument schemes similar to the manually implemented rules in [5]. Then we demonstrate how the ILP-derived rules could be used to model a reader's incremental interpretation of the arguments in the text from beginning to end. An incremental approach to extracting arguments is necessary since in some cases, an argument's conclusion was not explicitly stated in the genetics paper but functioned as a premise of subsequent arguments in the text. Another contribution of this paper is to present a computational model of the narrative of scientific discovery in that genetics paper.

The next section gives background on the annotation of arguments in the genetics paper. Section 3 describes the use of ILP to derive argument schemes from the annotated genetics paper. Section 4 describes the process used to simulate the reader's interpretation of the arguments in the text. Section 5 presents a computational model of the discovery narrative. Section 6 covers related work. Section 7 discusses the implications of this work and avenues for future research.

CMNA'21: Workshop on Computational Models of Natural Argument, September 2–3, 2021, Online
EMAIL: nlgreen@uncg.edu (N. Green); ljcrotts@uncg.edu (LJ. Crotts)

2. Annotated Genetics Article

Based upon the analysis of argument schemes in [9], we annotated the Results/Discussion section in a copy of [19].[2] After each segment of text, new discourse entities and/or propositions were annotated manually. The predicates used to encode the propositions in the text are as follows.

- have_genotype(I, G): Individuals I have genetic alteration (genotype) G
- have_phenotype(I, P): Individuals I have genetic condition (phenotype) P
- have_protein(I, P): Individuals I have protein P
- cause(G, P, I): Genotype G causes phenotype P in individuals I.

Also, propositions expressing the reader's presumed domain knowledge used in an argument are annotated using the following predicates.

- cognate(T1, T2): T1 is a cognate genotype of T2
- complement(I1, I2): Individuals I1 may be compared experimentally to individuals I2
- difference(A, B, C): The difference between genotypes A and B is genotype C
- isa(G1, G2): Genotype G1 is a subtype of G2
- similar(P1, P2): Phenotype P1 is similar to phenotype P2
- subset(I1, I2): Individuals I1 are a subset of individuals I2.

Then, any arguments in that segment were annotated. For example, the segment containing "…we analyzed genome-wide SNP data … from 577 individuals of European descent who were [controls] … We failed to find any deletions affecting the coding sequence of either gene, ITPR1 or SUMF1 …," was analyzed as introducing a new discourse entity for the control group, group6.[3] In a previous segment, it was asserted that another group of individuals, group5, had a certain genotype, geno5, and a certain phenotype, pheno3. In this discourse segment it is asserted that it is known that group6 does not have that genotype or phenotype. Also, we assume that a reader would have the domain knowledge that group5 and group6 can be compared in an experiment. The implicit conclusion of this argument, which is based upon Mill's Method of Difference, is that genotype geno5 may be the cause of the phenotype pheno3. This argument was annotated as shown at the top of Figure 1.

All of the arguments that were annotated had a conclusion that a certain genotype caused a certain phenotype or that that conclusion was known not to be the case. In addition to the Difference scheme illustrated above, other schemes include Agreement (based upon Mills' Method of Agreement), arguments based upon consistency, and a variant of argument from analogy with a causal conclusion.

3. Use of ILP to Derive Argument Schemes

An ILP problem is formulated as follows [1]. Given

(1) A set of positive examples E^+ and negative examples E^-,

(2) Background knowledge BK such that E^- cannot be derived from BK.

Find a hypothesis H such that

(1) all of the examples in E^+ can be derived from BK and H, and

(2) none of the negative examples in E^- can be derived from BK and H,

where the examples, background knowledge and hypothesis are logic programs.

In this project, we have used the ILP system, CProgol 4.4 [14]. The background knowledge given as input to Progol included all manually annotated propositions from the genetics paper and presumed domain knowledge of the reader, using the predicates listed in the previous section. The fifteen annotated argument instances from the genetics article were encoded as positive examples. (No negative examples were provided and are not required by Progol.) Ten rules were derived by Progol. A positive example of an argument that was input to Progol is shown in part II of Figure 1.[4] The premises are described by identifiers of the corresponding propositions in the background knowledge,

[2] The annotated article is available at https://github.com/greennl/BIO-Arg-ILP. The annotation scheme is described in [6].
[3] Entity identifiers such as group6, geno5, pheno3, etc. are assigned by the annotator.
[4] The full input to Progol can be seen at https://github.com/greennl/BIO-Arg-ILP.

i.e., the premises are listed as id26, id27, id28, id30, and id67. The propositions with those identifiers are shown in part III of Figure 1. In other words, the premises are (id26) that the affected members of a certain family (group5) have a phenotype pheno3 (a genetic condition referred to as SCA15) and (id27) they have a genotype geno5 (a deletion in the region ITPR1-SUMF1), and (id67) a control group (group6) is (id 28) known not (*knot*) to have genotype geno5 nor (id30) phenotype pheno3. (Propositions id26, id27, id28 and id30 came from the annotated text; proposition id67 came from domain knowledge.) The conclusion is that geno5 may be the cause of pheno3 in group5.

I. Annotated argument:

```
<argument scheme="Difference">
<premise-list>
<premise prop="have_pheno(group5, pheno3)" />
<premise prop="have_geno(group5, geno5)" />
<premise prop="knot(have_pheno(group6, pheno3))" />
<premise prop="knot(have_geno(group6, geno5))" />
<premise domain-prop="complement(group5, group6)" />
</premise_list>
<conclusion inferred-prop="cause(geno5, pheno3, group5)" />
</argument>
```

II. Encoding of the same argument input to Progol as a positive example:

```
argument7(difference, id26, id27, id28, id30, id67, conclusion(cause(geno5, pheno3, group5))).
```

III. Related background knowledge input to Progol:

```
has_pheno(id26, group5, pheno3).                Members of family AUS1 have SCA15
has_geno(id27, group5, geno5).                  Members of family AUS1 have ITPR1-SUMF1 deletion
knot(id28, has_pheno(id29, group6, pheno3)).    Control group is known not to have SCA15
knot(id30, has_geno(id31, group6, geno5)).        and is known not to have ITPR1-SUMF1 deletion
complement(id67, group5, group6).               Experimentally comparable groups
```

IV. Rule acquired by Progol for the above argument instance:

```
argument7(difference, A, B, C, D, E, conclusion(cause(F,G,H))) :-
        has_geno(B,H,F),                B: group H has genotype F
        has_pheno(A,H,G),               A: group H has phenotype G
        knot(C,has_pheno(I,J,G)),       C: it is known not I (group J has phenotype G)
        knot(D,has_geno(K,J,F)),        D: it is known not K (group J has genotype F)
        complement(E,H,J).              E:  group H is the complement of group J
```

Figure 1: Sample of input to Progol and derived rule (with comments in italics).

The rule acquired by Progol describing that argument instance (as well as the other three instances of that type of argument) is shown in part IV of Figure 1. The predicate name argument7 at the head of the rule refers to the arity of the predicate. The value of the first parameter in the rule head, "difference", is the name assigned to this argument scheme.[5] The variables[6] A, B, C, D, and E in the head refer to the premises of the argument, and are identifiers of propositions in the body of the rule. The conclusion of the argument is that genotype F may be the cause of phenotype G in group H.

[5] In addition to being useful for purposes of documentation, including a scheme name can be thought of as a placeholder for linking the scheme to its critical questions in future work.

[6] In Prolog, terms that begin with an uppercase letter are variables.

Progol finds the most specific clause that entails a positive example, using mode declarations to supply possible predicates in the body of the rule. Progol tries to generalize the most specific clauses by eliminating predicates from the body that are not needed to cover the positive examples or to rule out negative examples. However, we have used only the most specific clauses created by Progol since the generalized rules omitted necessary premises (according to domain experts), generating unacceptable arguments. Thus, it may turn out not to be feasible to use ILP to acquire generalized argument schemes that reflect scientifically acceptable arguments. If that is the case, manually encoded logic-programming rules such as presented in [5], reflecting scientists' judgements of acceptability, could still play a valuable role in argument mining the scientific literature.

4. Modeling the Reader's Interpretation of Arguments

As a demonstration of how the Progol-derived argument scheme rules could be used, the following simulation of a reader's incremental interpretation of the arguments in the text was performed to extract the premises and conclusion of each argument. The argument scheme rules derived by Progol and the reader's presumed domain knowledge were asserted into Prolog's knowledge base at the beginning of the simulation. Then, for each annotated discourse segment of the text, its annotated propositions were asserted into Prolog's knowledge base. After they were asserted, each argument scheme rule was run as a Prolog query. For each successful query, all argument instances were recorded and the conclusions of the arguments were asserted into the knowledge base. The conclusions were asserted since, as mentioned above, not all of the arguments' conclusions were stated explicitly in the text, and some of the asserted conclusions are needed as premises of arguments in subsequent segments.

None of the annotated argument instances failed to be derived by the rules, and no unacceptable argument instances were returned. However, in an earlier version of the simulation, certain argument schemes were applied inappropriately in certain contexts. To address this situation, we added contextual propositions as text annotations using the following two predicates:

- enable(Scheme): Argument Scheme may be used in the current discourse context
- focus(Species1, Species2): Both species (e.g. mice and humans) are in focus.

The proposition *focus(mouse, human)* was added to the annotation of discourse segments in which data from mice and data from humans were used in an argument. The same focus proposition was added as a premise to the positive examples of argument schemes allowing those comparisons, e.g. Consistent Explanation and Analogy. In a subsequent segment where such a comparison was not allowed, the annotation of the text specified that the focus should be retracted. Also, the *enable(Scheme)* predicate was used to allow the application of the named Scheme only in contexts in which it was appropriate. In this text, arguments by Analogy were used only to generate hypotheses for further investigation and are weaker than the other types of arguments used. Thus, the proposition *enable(analogy)* was added to the annotation of discourse segments in which the Analogy argument type was allowable and was added as a premise to the positive examples of Analogy. In a subsequent segment where that type of argument was not allowed, the text annotation specified that the enabling of the scheme should be retracted.[7] An example of the text annotation of an argument by Analogy, incorporating those premises, is shown in Figure 2.

In addition to the annotated argument instances, the rules derived a number of acceptable arguments that had not been annotated manually. In some cases, the conclusion of the unannotated argument found by the rule was the same as the conclusion found by another rule but was based upon different premises. In some of those cases, the premises of one argument scheme (e.g. Agreement) were a subset of the premises of another scheme (e.g. Difference). Another case where the derived rules returned unannotated arguments was when the article had cited evidence from previous studies and the rules provided arguments for the conclusions of those studies. Finally, in some cases the rules derived arguments for subgroups of individuals that were subsumed by arguments for groups containing those subgroups.

[7] Analogy was the only type of argument for which both *focus* and *enable* propositions were required.

```
<argument scheme=”Analogy”>
<premise-list>
<premise prop=”enable(analogy)” />
<premise prop=”have_pheno(group3s, pheno2)” />
<premise prop=”have_geno(group3s, geno2a)” />
<premise domain-prop=”similar(pheno2, pheno3)” />
<premise prop=”cause(geno2a, pheno2, group3s)” />
<premise domain-prop=”cognate(geno2a, geno4)” />
<premise prop=”have_pheno(group4, pheno3)” />
<premise prop=”focus(mouse, human)” />
</premise_list>
<conclusion prop=”cause(geno4, pheno3, group4)” />
</argument>
```

Figure 2: Annotation in text of an argument by analogy

5. Modeling the Discovery Narrative

An interesting question is how to describe the relationships among conclusions of the arguments in the genetics article. In [5] we noted that a model of argument attacks did not accurately describe this genetics article. In the article, no conflicting arguments were made. We noted that, somewhat like a discovery dialogue in multi-agent systems [11], the goal was to discover previously unknown knowledge. The authors of the genetics article make the case for their final conclusion by presenting a narrative of the steps they took to reach it (observing a genetic condition in some mice in their lab, running experiments to determine its cause, hypothesizing an analogous mutation and condition in humans, running experiments to determine the cause in humans, etc.).

Abstracting away from these reported activities, it is possible to represent the narrative by a finite state automaton as shown in Figure 3. The states represent conclusions of annotated arguments in the article, e.g., state S_1 is the conclusion of argument 1 that a mutation on chromosome 6qE1 is the cause of a disorder observed in some mice in the authors' lab.[8] (The conclusion of each argument is that a certain genotype is the cause of a certain phenotype in a certain group of individuals.) States with multiple subscripts, such as state $S_{3\text{-}4}$, represent states with the same conclusion (i.e., where multiple arguments for the same conclusion were given). Arcs are annotated with the relationship between states. S_i *refine* S_{i+1} indicates that the cause in S_{i+1} is a specialization of the cause in S_i, S_i *isa* S_{i+1} indicates that the cause in S_{i+1} is a generalization of the cause in S_i, S_i *confirm* S_{i+1} indicates that the cause in S_{i+1} is the same as the cause in S_i, and S_i *cognate* S_{i+1} indicates that the causal relation in S_{i+1} is a transspecies cognate of the cause in S_i. The states in Fig. 3 are annotated with the population to which the causal statement applies: humans with SCA15, mice in the authors' lab with ataxia (found to have the Itpr1 del18/del18 genotype), mice described in a previous study (with the Itpr1 opt/opt genotype), AUS1 (affected members of one human family with SCA15), H33/H27 (affected members of two other human families with SCA15).

Some of the conclusions in Fig. 3 entail conclusions of other arguments but are not themselves entailed by other arguments. In order to prune the Prolog knowledge base, at the end of the simulation, we implemented a few update rules in Prolog that retract the unneeded conclusions, leaving only the conclusions in the nodes labeled Previous, $S_{3\text{-}4}$, S_{4S}, S_{10}, S_{11}, and $S_{12\text{-}14}$. The final conclusion of the article ($S_{12\text{-}14}$) is that the cause of the genetic condition SCA15 in humans is an ITPR1 deletion.

[8] The node labelled *Previous* represents a conclusion of previous research cited in the article.

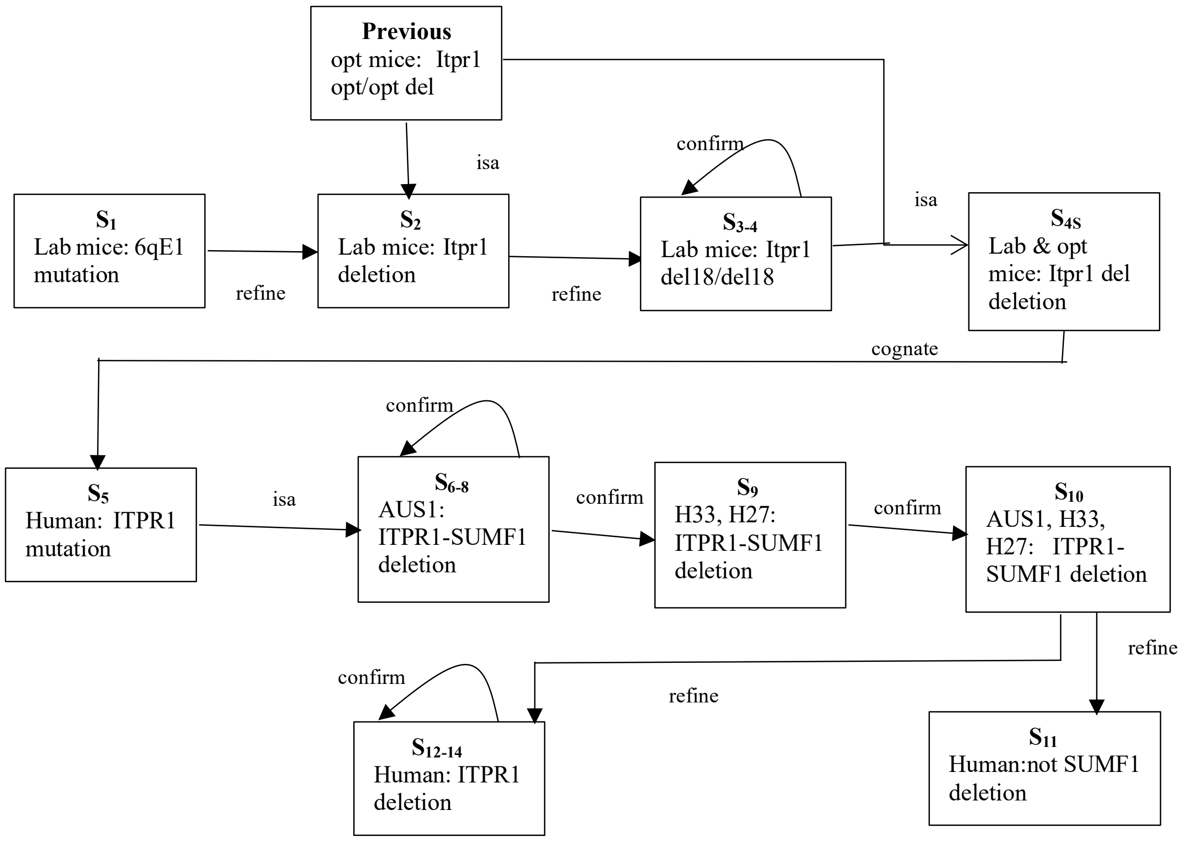

Figure 3: Flow of conclusions in the genetics article.

6. Related Work

Moser and Mercer [12] report on a manual analysis of five biochemistry journal articles that showed that the argument schemes that we proposed for genetics research articles[9] are applicable to their subject and suggested that the schemes may be applicable to the experimental biomedical literature in general. (Note that we do not claim that the set of schemes we identified is exhaustive.) In addition, they developed claim graphing, a manual technique for understanding the argumentation structure of a complete biochemistry paper. Claim graphs show the support relationships among claims, as well as between claims and data contained in figures and tables. They propose that in the future automatically constructed claim graphs could be used in automatic summarization to identify the most important claims. Although differing in details, claim graphs have a similar purpose to that of the model of the discovery narrative presented in the previous section, i.e., to provide a summary of the argumentation and to identify the most important claim(s).

In formal argumentation, abstract bipolar argumentation frameworks (BAF) [3] have been proposed as an extension to abstract argumentation frameworks. The motivation for representation as a BAF is to determine the strength of various arguments in order to decide which claim(s) to accept. In addition to attack relationships, a BAF models independent support relationships between arguments, e.g., when the conclusion of one argument functions as a premise of another argument. It is an open question how the strength of claims in our model of a discovery narrative could be represented in a BAF.

[9] They used an earlier version of the scheme definitions that were not expressed in terms of genetics concepts.

There has been little work on machine learning of argument schemes (and none in the biological/biomedical sciences). Feng and Hirst [4] applied classifiers to non-semantic features to recognize occurrences of five common argumentation schemes described by Walton et al. [20] in the Araucaria corpus, in which premises and conclusions of arguments had been previously annotated. Lawrence and Reed [8] attempted to infer some of the Walton et al. catalogue of argumentation schemes in a corpus of arguments extracted from a 19th century philosophy text, in which premises and conclusions had been previously annotated along with limited semantic information about the type of a premise. There has been little use of logic programming for argument mining. Saint-Dizier [16] used rules manually encoded in a logic programming language for automatic identification of arguments in instructional/opinion texts. However, unlike in our approach, the rules are based on syntactic patterns and lexical features. In later work [17] he used an enriched lexicon as a supplement to the text for argument mining. Most previous argument mining researchers have used statistical, non-semantics-informed machine learning approaches on corpora not derived from the biological/biomedical sciences [9, 10, 18]. Inductive logic programming has been used for scientific discovery and NLP [2, 7].

7. Discussion

This paper has described how argument schemes were derived from a semantically-annotated and argument-annotated genetics research article by ILP. The schemes were used successfully to extract premises and conclusions of arguments, including arguments whose conclusions were not stated explicitly in the text. Of course, as a preliminary test of this proposed approach, it has considerable limitations. Due to a lack of resources, we have been unable to annotate, train on, and test on more than one genetics article. (In the near future, we plan to continue annotation and testing on additional genetics articles.) Furthermore, it would be ideal to test this approach on articles whose entities and relations had been automatically recognized by BioNLP tools rather than human annotators. Also, future work could explore approaches to ILP that are tailored to this type of application. Nevertheless, this is the first work that we are aware of that has attempted to mine arguments in the natural sciences research literature using semantics-informed, learned argument schemes. We were pleasantly surprised that the rules extracted acceptable arguments that we had not recognized during the initial annotation effort. It will be interesting to see what arguments this initial set of rules can extract in other works in this domain.

In addition, this paper presented a computational model of a discovery narrative built on the conclusions of extracted arguments in a genetics article. Future work could address use of such a model in automatic summarization.

8. References

[1] Bratko I. Prolog Programming for Artificial Intelligence, 3rd ed. Addison-Wesley, 2001.

[2] Bratko I, Muggleton S. Applications of inductive logic programming. Communications of the ACM 1995; 38(11): 65–70.

[3] Cayrol C, Lagasquie-Schiex M-C. Bipolar abstract argumentation systems. In I Rahwan, GR Simari, Argumentation in Artificial Intelligence. Springer, 2009, 65-84.

[4] Feng VW, Hirst G. Classifying arguments by scheme. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, Portland (OR), 2011; 987–996.

[5] Green NL. Towards Mining Scientific Discourse Using Argumentation Schemes. Argument and Computation 2018; 9(2):121-135.

[6] Green NL. Proposed Method for Annotation of Scientific Arguments in Terms of Semantic Relations and Argument Schemes. Proc. Fifth Workshop on Argumentation Mining, Nov. 1, 2018, Association for Computational Linguistics.

[7] Gulwani S, Hernandez-Orallo J, Kitzelmann E, Muggleton S, Schmid U, Zorn B. Inductive programming meets the real world. Communications of the ACM 2015; 8(11): 90–99.

[8] Lawrence J, Reed C. Argument mining using argumentation scheme structures. Computational Models of Argument: Proceedings of COMMA 2016, IOS Press, Amsterdam; 379–390.

[9] Lawrence J, Reed C. Argument Mining: A Survey. Computational Linguistics 2019; 45(4): 765-818.

[10] Lippi M, Torroni P. Argumentation mining: State of the art and emerging trends. ACM Transactions on Internet Technology 2016; 16(2):10.

[11] McBurney P., Parsons S. Chance discovery using dialectical argumentation. New Frontiers in Artificial Intelligence, T. Terano et al., eds, Lecture Notes in Artificial Intelligence, v.2253, Springer Verlag, Berlin, 2001, pp. 414–424.

[12] Moser E, Mercer RE. Use of Claim Graphing and Argumentation Schemes in Biomedical Literature: A Manual Approach to Analysis. Proc. of the 7th Workshop on Argument Mining. ACL, 2020; 88-99.

[13] Muggleton S, De Raedt L. Inductive Logic Programming: Theory and Methods. Journal of Logic Programming 1994; 19, 20: 629-679.

[14] Muggleton S, Firth J. CProgol4.4: a tutorial introduction. 2001. Downloaded from: https://www.doc.ic.ac.uk/~shm/Papers/progtuttheo.pdf

[15] Perera N, Dehmer M, Emmert-Streib F. Named Entity Recognition and Relation Detection for Biomedical Information Extraction. Frontiers in Cell and Developmental Biology 2020; 8. doi: 10.3389/fcell.2020.00673.

[16] Saint-Dizier P. Processing natural language arguments with the<TextCoop> platform. Argument & Computation 2012; 3(1):49–82.

[17] Saint-Dizier P. Knowledge-driven argument mining based on the Qualia structure. Argument and Computation 2017; 8(2):193–210.

[18] Stede M, Schneider J. Argumentation Mining. Synthesis Lectures on Human Language Technologies. Morgan and Claypool; 2018.

[19] Van de Leemput J, Chandran J, Knight M, et al.. Deletion at ITPR1 underlies ataxia in mice and spinocerebellar ataxia 15 in humans. PLoS Genetics e108 2007: 1076–1082.

[20] Walton D, Reed C, Macagno F. Argumentation Schemes. Cambridge University Press; 2008.