Explanation-based Learning Helps Acquire Knowledge from Natural Language Texts

Sylvain Delisle, Stan Matwin, Jiandong Wang, Lionel Zupan[1]
Department of Computer Science, University of Ottawa
Ottawa, Ontario, K1N 6N5, Canada
Tel: (613) 564-5069, email: stan@csi.uottawa.ca

ABSTRACT

Existing systems to acquire knowledge from expository texts do not perform any learning beyond interpreting the contents of the text. The opportunity to learn from examples included in texts is not exploited. This is a needless limitation because examples in texts usually show the reader how to integrate the declarative part of the text into an operational concept or procedure. Explanation-based Learning (EBL) seems to fill this gap as it explains the example within the domain theory, generalizes the explanation and operationalizes the concept definition by compiling necessary knowledge from the domain theory into the definition. In this paper, we study the synergistic combination of automatic text analysis and EBL. EBL is used realistically, where the domain theory and the training examples are obtained from a specification or a regulation by a text analysis program, rather than being given *a priori*. We present a prototype system which demonstrates the potential of this approach. The paper includes a detailed example using the Canadian Income Tax Guide.

1. Introduction

1.1 Automatic Knowledge Acquisition from Text

Manuals, textbooks, specifications, and regulations (often referred to as *expository* texts) are a fundamental source of human knowledge. These texts are often the primary sources of expertise for knowledge-based systems. Consequently, computer programs that help transfer knowledge from texts to knowledge bases are an important knowledge acquisition (KA) tool. While research on KA from texts traditionally was carried out in the field of natural language processing (NLP), recently a consensus is forming that any such tools will have to draw upon machine learning methods to generalize, compile and make knowledge operational. In a recent meeting of the Machine Learning community [Austin 90], the problem of KA from texts has been put forward as one of the challenges that the ML field should address. This paper focuses on the automatic acquisition of knowledge from texts that include examples. We show how the problem of integrating knowledge contained in the expository text with the knowledge contained in the examples can be solved using ML techniques.

Expository texts usually have two components: 1) a declarative presentation of the domain knowledge in a narrative form, and 2) specific examples of individual objects, processes, and procedures in the domain described by the text. A number of *text analysis systems* (TAS), i.e., automatic systems for KA from texts (e.g., [Frey et al. 83], [Gomez 89], [Moulin & Rousseau 90], [Nishida et al. 86], [Regoczei & Hirst 89], [Reimer 90], [Rinaldo 89], [Silvestro 88], [Szpakowicz 90]) work upon expository texts. These existing TAS have difficulty processing examples. In fact, as far as we know, all existing TAS simply ignore

[1] Ecole Nationale Supérieure des Télécommunications, Paris, France.

examples. There are many reasons for not analyzing examples. Representation and the language in which examples are described is often drastically different from that adapted for the narrative text: the former may be a diagram, a figure or a program, while the latter is given in natural language. Moreover, even when the two representations are consistent, the languages in which the two components of expository texts are presented differ in their lexical and syntactic definitions. It is the concepts underlying both languages that link the two components. As the first step towards a general solution of the problem of processing examples in KA, we treat the case in which the examples are in a format compatible with the text.

1.2 Bridging the Gap between Declarative and Example-Based Knowledge Acquisition

Not using examples to the fullest extent in KA from text is certainly a drawback. Expository texts rely on examples to show the reader how to integrate different rules contained in the declarative part of the text into an operational concept or procedure. The writers of expository texts often expect their readers to use examples to generate reasonable abstractions for appropriately using a concept or procedure.

Explanation-Based Learning (EBL) ([Ellman 89], [Mitchell et al. 86], [DeJong & Mooney 86]) seems to fill perfectly the gap in the existing TAS. EBL explains the example within the domain theory, and compiles, into the example itself, parts of the domain theory necessary to operationalize the example. Moreover, the explanation and its operational part are generalized, so that the result is usable not only for the single example but for any case that has the same justification.

This paper studies the synergistic connection between TAS and EBL. Using an existing, simple parser included in a TAS, and an implementation of EBG, we have built a prototype of an integrated system that performs both EBL and TAS. Our prototype system for the INtegration of Text analysis with ExpLanation-based LeArning is called INTELLA. Early experience with INTELLA indicates that our approach actually increases the knowledge acquired by TAS alone. As well, the EBL that INTELLA performs would have been difficult, if not impossible, if the components of EBG were not provided by a TAS. Consequently, we have shown that a fusion of TAS and EBL is a synergistic KA tool, in that it produces more knowledge than either of its two constituents.

1.3 Related Work

The problem we have presented above is raised and discussed in [Cohen 90]. It must be observed, however, that the treatment presented there does not address the main practical problems. The approach of [Cohen 90] assumes that a text analysis program exists, and is limited to manual text translation of natural language into Horn clauses. The focus in that paper is more on using examples from expository texts to identify gaps in an incomplete theory, than on learning new, operational rules. [Cohen 90] gives also a PAC-like characterization of the size of the sample necessary to learn the approximate rules.

The approach we propose here may be the first to consider examples as an intrinsic part of a text in view of acquiring knowledge from them during text analysis. Let us now briefly review some recent related projects which have focused mainly on the acquisition of knowledge from text, but which all exclude examples. The goal of these systems is automatic

construction of rules that represent domain knowledge. Typically, these rules are to be used by knowledge-based systems and, in particular, expert systems.

Several approaches have been proposed to engage in classification, which are typical of knowledge base construction activities. [Silvestro 88] presents an expert system tool, which he refers to as a knowledge-base acquisition mechanism, that constructs knowledge bases by accepting English explanations (i.e., explanatory text) from the user. Silvestro's approach seems to be restricted to taxonomic knowledge, based solely on the is-a relation (e.g., 'a course is an episode', 'a department is an organization'). Two somewhat similar approaches are those of [Gomez 89] and [Reimer 90] which are also restricted to classification tasks, although their output knowledge representation is that of conceptual structures instead of production rules.

A different approach is exemplified by the work of [Rinaldo 89] and [Moulin & Rousseau 90] in which text processing is reduced to the simplest expressions. Their systems do not perform any real (i.e., 'deep') semantic processing. Their simple surface-syntactic processors look for pre-determined and fixed patterns (e.g., 'if', 'because', 'when') in the input sentences, and use these key words to decompose the original sentences into representations that stand for production rules. In both systems, user intervention is necessary in order to insure that the rules produced (or suggested) automatically are relevant and correct.

2. Correspondence between EBL and examples in expository texts

We assume that the explanatory texts processed by TAS have the following components:

- the narrative text describing the domain
- examples, which illustrate rules, concepts, and procedures defined in the declarative part
- captions, which accompany examples and highlight their contents. Captions name the concept, or procedure, presented in the example, and often describe it in most general terms. They do not provide the details of the concept or procedure definition. Captions link the example with the narrative text, by listing the concepts that the example illustrates.

We propose to use EBL with the following mapping of parts of the expository text into data necessary for EBL:

- the narrative text is converted into a domain theory. This is done by a TAS, sometimes with the assistance of a human operator (e.g., in removing certain types of natural language ambiguities)
- examples of concepts or procedures from the expository text are used as EBL's training examples. These are translated by a TAS form NL into a format conducive to EBG
- captions are used as non-operational definitions of concepts and procedures

The result of EBL, i.e., the operationalized and generalized concept or procedure, is then added to the knowledge base that has been acquired by TAS from the expository text. Although the learning that has been achieved is of the "deductive" type [Dietterich 89], it enhances the knowledge base with new and potentially useful rules that cannot be obtained from the analysis of the declarative part alone. Section 4 contains an example of such a rule.

The particular text we have selected for experimenting with our approach is the Canadian Income Tax Guide. There are two good reasons for this choice: i) this is a non-trivial technical expository text about a well-structured domain which is described mainly in terms of rules and procedures, and ii) this text makes intensive use of examples. For now, we have decided to concentrate on examples to test our approach. In this way, we minimize the effort involved in analyzing the text of the domain theory, which happens to be linguistically much more complex than the examples (mainly because of the ubiquitous use of structurally ambiguous conjoined structures which are problematic for automatic text processing systems). Nevertheless, we plan to consider automatic acquisition of the domain theory soon.

3. A Prototype System Integrating Text Analysis with Explanation-based Learning

This section describes our approach to knowledge acquisition from examples written in natural language. Our approach contrasts with most EBL applications where the knowledge to be processed by the EBL engine is assumed to be already available in first-order logic (or an equivalent representation, e.g., [Cohen 90]). The method presented here does not make such an assumption: we begin with the text in its original natural language form. From this text we produce the appropriate knowledge representation which, in turn, feeds the EBL engine. Figure 1 (below) represents the current architecture of the system. The remaining part of this section details the system's components.

3.1 The Natural Language Simplifier

This module takes as input the original text that constitutes the training example and automatically performs syntactic simplifications on those sentences that are too complex for the parser. The simplifications include decomposing of complicated conjoined structures, decomposing of enumerations, simplifying of verbal forms, and several other procedures. Some of these simplifications, especially for conjoined structures, are similar to those performed by the Regularization Modules [Friedman 86] of the Linguistic String Project [Sager 81] (see also the process of linearization in [Fong & Berwick 85]).

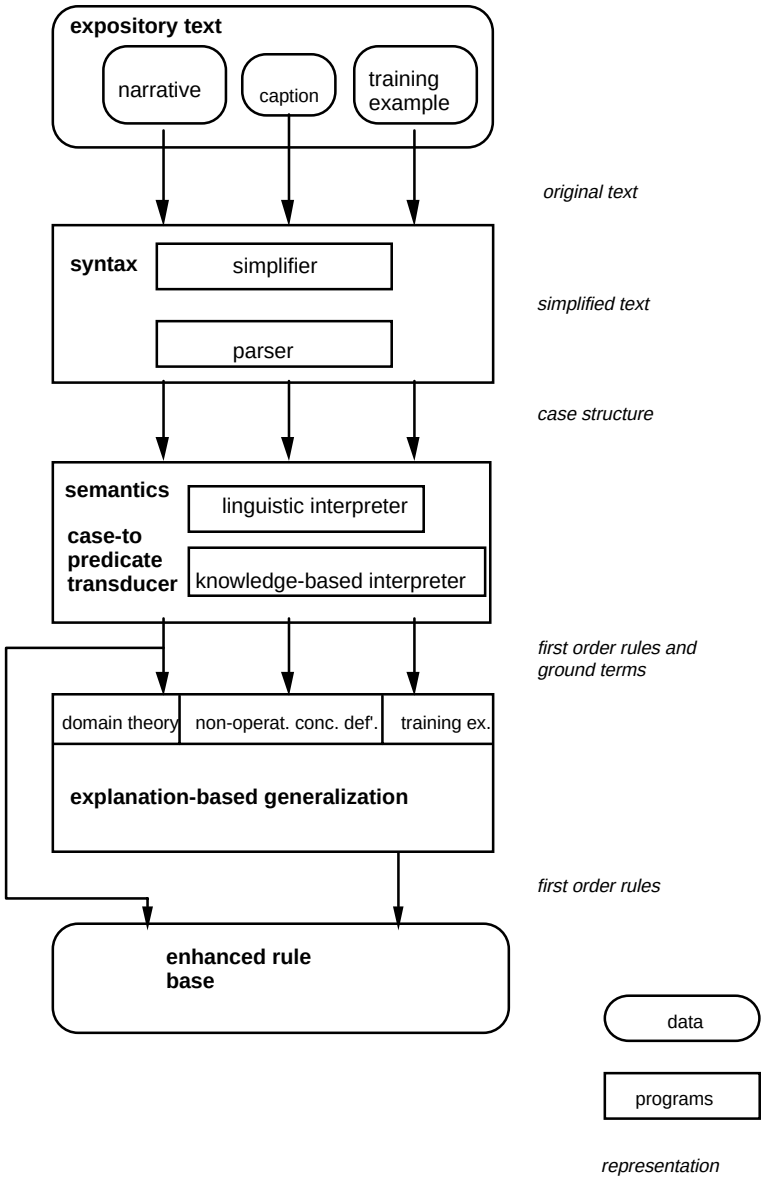


Fig. 1. General architecture of INTELLA

Because simplifications are performed on purely syntactic grounds, they sometimes produce results that are not acceptable to humans. This is an intrinsic limitation of simplification rules, which tend to over-generalize. For sentences that deserve a better simplified form, the operator is expected to step in and suggest a new simplified sentence that should be parsable by the next module. Up to now, we have been able to reduce this participation to a minimum. The current set of simplifications correctly covers almost all situations encountered in the examples we have considered so far.

At the time of writing, the NL Simplifier is implemented in Prolog and has been tested with success on three examples from the Tax Guide domain. Each example contains between five and ten sentences.

3.2 The Parser

Once the original input text has been processed by the NL Simplifier, the result is passed on to the parser which analyzes it one sentence at a time. The parser is a slightly modified version of SEQAP [Delisle 87], a Case-based parser originally designed and implemented as a natural language front-end to a prototype expert advisor system. The output of the parser is, for every sentence submitted on input, a case structure [Bruce 75, Somers 87] in which the main verb and its arguments represent the essence of the input sentence.

As mentioned at the end of sec. 2, we also plan to apply our approach to the (semi-) automatic acquisition of the domain theory. To that end, we are considering replacing SEQAP with a much more powerful parser [Delisle 90], which is a component of a text processing research project currently in progress [Szpakowicz 90, Copeck et al. 90].

3.3 The Case-to-Predicate Transducer

The parser produces case structures and the EBL engine expects simple Prolog facts on input (instead of Prolog Horn clauses.) We must bridge the gap between the two with a transducer that converts the parser's output into a form amenable to EBL processing. Conversion entails several problems on which we are currently working. One of these problems is to select the most appropriate predicate name for the case (sub)structure associated with a given sentence. This problem is generally a difficulty in first-order-based knowledge representation derived from natural language utterances, for which there exist only limited solutions.

Our approach in implementing the transducer was to divide the translation in two levels: domain-independent, purely syntactic interpretation, and a domain-specific, knowledge-based interpretation. We also designed the transducer as a definite clause grammar (DCG). Using grammar rules seems to be the most appropriate strategy to implement the Transducer, because: i) the parser's output can be adequately described with a DCG, and ii) such a rule-based approach permits easy future enhancements of the Transducer, by just adding new grammar rules.

The first level is purely linguistic and automatically produces Prolog facts from the parser's output, e.g.,

> Original sentence: "Adam goes to university."
> Linguistic level: go(adam,university).

At the time of writing, a simple anaphora resolution mechanism is also implemented. It allows to keep track of some intersentential relations. A stack is progressively built during the one-at-a-time sentence treatment of the Transducer. Each time a pronoun occurs, the Transducer refers to the stack and tries to determine the reference by considering compatibility criteria:

Original sentence:	"Adam.... His parents pay ...."
Anaphora resolution:	parent(Adam,Parents), pay(Parents,...).

The second level of the transducer consists of a domain-specific translation of the Prolog facts produced by the first level:

Linguistic level:	go(adam,university).
Knowledge-based, domain-specific level:	student(adam).

This level is implemented by demon-like rules that are enacted whenever a given predicate created by the first level is encountered. We can observe that the domain specific, knowledge-based interpretation of predicates is to be circumscribed by the context. For instance, it is appropriate to interpret go(adam,university) as student(adam) in the context of a domain theory on taxation, but not necessarily in the context of a domain theory on commuting.

3.4 EBL Engine

In INTELLA, we rely on an Explanation-Based Generalization (EBG) engine to generalize from examples. The main extension to standard EBG involves an enhancement of the operationality criterion, so that EBG produces results at different levels of abstraction. In order to be considered operational, a node must be useful and directly interpretable by the agent, be it a person or a system component. Unlike the rules from the domain theory that built the explanation tree above the operational level, the rules below the operational level rely on simple, rather than in-depth knowledge (see example in sec.4.) Such an extension gives an agent an operational concept definition, collected in the nodes of the tree that are labelled "operational". If the example lacks the information required to continue the proof below the level of operationality, then a standard EBG explanation is obtained. The generalization is then produced more abstractly, leaving out some details about the procedure used, for example, to evaluate some arguments. For instance, the example in sec. 4 does not provide the specific values for calculating the unused tuition fees of a given person, so the generalization gives the information of "determine_deduction(TFEA$[2], OTC$, FIT$, UTFEA$)"[3]. If, however, the values in the example are provided, the deductive explanation will continue and a procedure to compute the UTFEA$ will be given, such as if TFEA$+OTC$>FIT$ and OTC$>FIT$, UTFEA$ is equal to TFEA$ (see Fig. 2).

The two levels of rules - above and below the operationality level - are consistent with two types of knowledge involved in EBL: domain theory and background knowledge. The former is specialized knowledge, usually unfamiliar to the reader of the expository text. The latter is the usual background, assumed of any reader in the group to which the expository text is addressed. In the current phase of work on INTELLA, the domain theory is coded manually. This initial approach responds to the syntactic complexity of the expository text devoted to presenting the domain theory. With the availability of a more powerful parser (sec. 3.2), we plan to automate this step.

Background knowledge, on the other hand, always has to be provided manually, as it is not part of the expository text. Future availability of general, repository knowledge bases (e.g.,

---

[2] Each variable ended by a '$' represents an amount of money.

[3] In this example, [U]TFEA means "[Unused] Tuition Fees and Education Amount". FIT,TNRTC and OTC mean "Federal Income Tax","Total Non-Refundable Tax Credits" and "Other Tax Credits", respectively.

[Lenat et al. 90]) will alleviate this problem. In the current implementation, both the domain theory and the background knowledge are represented in the same Prolog rule format, and are indistinguishable by the EBL engine.

The components of EBL are illustrated with the example discussed in the next section. The results presented in that section are produced by an implemented EBG engine.

4. INTELLA at Work on the Canadian Income Tax Guide

In this section, we present in more detail an example of INTELLA at work. We show, in turn, part of the source text for the domain theory, a fragment of the theory represented as Prolog rules, the source text of the training example, the automatically simplified text of the training example, the explanation of the example using the domain theory and background knowledge, and finally the operational result of learning.

4.1 Domain theory

The chosen example is extracted from the section of the Canadian Income Tax Guide devoted to the transfer of "tuition fees and education amount" from a child to a parent. This knowledge is referred to as "Line 324", from the line in the tax return form in which such a deduction may be claimed. For the sake of brevity, we show here only the first sentence of Line 324:

> " If a student does not need to claim tuition fees and the education amount, or either, to reduce federal income to zero,the unused part (to a maximum of $3,529 for each student) may be transferred to the student's parent or grandparent (including in-laws)."

The entire domain theory pertaining to Line 324 that can be acquired from the Tax Guide is shown below. We have used a notation in which TaxPayer is the person who "declares" and Claimant (or OtherPerson) is the person who "receives" the transfer or "claims" it:

```
transfer(TaxPayer,unused_part_tuition_fees_and_education_amount,
         TFEA$, Max_UTFEA$, Claimant) :-
         student(TaxPayer),
         does_not_need_tfea(TaxPayer, TFEA$, UTFEA$),
         meet_conditions(Claimant, TaxPayer)[4],
         is_the_only_claimant(Claimant)[5],
         calculate_max_utfea(UTFEA$, Max_UTFEA$).

meet_conditions(Claimant, TaxPayer) :-
         married(Claimant, TaxPayer),
         makes_claim(Claimant, Kind_of_claim, TaxPayer),
         claim(Claimant, transfer_tuition_fees_and_education_amount,
               _, TaxPayer).
```

[4] `meet_conditions/2`, `is_the_only_claimant/1` and `makes_claim/3` were deduced from the rest of Line 324 that does not appear above.

[5] We will not detail this predicate: it succeeds if Claimant is the only claimant or fails if not.

```
meet_conditions(Claimant, TaxPayer) :-
        relative(Claimant, TaxPayer), dependant(TaxPayer, Claimant).
```

```
meet_conditions(Claimant, TaxPayer) :-
         not (dependant(TaxPayer, _)), relative(Claimant, TaxPayer),
         claim(Claimant, transfer_tuition_fees_and_education_amount,
               _, TaxPayer),
         not (married(OtherPerson, TaxPayer),
               makes_claim(OtherPerson, Kind_of_claim, TaxPayer)).

makes_claim(_, married_amount, _).
makes_claim(_, amounts_transferred_from_spouse, _).

does_not_need_tfea(TaxPayer, TFEA$, UTFEA$) :-
         get_from_line_401(TaxPayer, FIT$),
         get_from_line_350(TaxPayer, TNRTC$),
         OTC$ = TNRTC$ - TFEA$,
         determine_deduction(TFEA$, OTC$, FIT$, UTFEA$).

determine_deduction(TFEA$, OTC$, FIT$, TFEA$) :-
         TFEA$ + OTC$ > FIT$, OTC$ > FIT$.
determine_deduction(TFEA$, OTC$, FIT$, X) :-
         TFEA$ + OTC$ > FIT$, X is TFEA$ + OTC$ - FIT$.
determine_deduction(_, _, _, 0).

calculate_max_utfea(X$, X$) :- X$ < 3539.
calculate_max_utfea(X$, 3529) :- X$ >= 3529.
```

4.2 Training Example

Below, we are quoting the training example exactly as it is given in Line 324:

> "Adam is 19 years old, goes to university and has no income.His parents paid his tuition fees for courses he took in 1989.As Adam does not have any income,he may transfer all of his 'tuition fees and education amount' to one of his parents."

As mentioned in sec. 3.1, we automatically preprocess the original, unconstrained text to remove conjunctive verb phrases and object-related relative clauses. In the next stage, the Parser and the Case-to-Predicate Transducer convert the simplified text, obtained by NL Simplifier (without any supervisor's intervention), into the following Prolog facts:

```
age(adam, 19).
student(adam).
income(adam, 0).
parent(Y, adam).
pay(Y, tuition_fees_and_education_amount, TFEA$, adam).
transfer(adam, tuition_fees_and_education_amount, TFEA$, Y).
```

Observe that the transducer makes the semantic conversion of the verb `to go` as occurring in the given context in the example into the predicate `student(adam)`. This is described in more detail in sec. 3.3.

4.3 Background Knowledge

In order to be able to explain any facts in the tax domain, INTELLA had to be given the basic knowledge of what is a parent or grandparent, what it means to be a dependant for tax purposes:

```
dependant(Y,X) :- parent(X,Y), age(Y,A), A < 18.
dependant(Y,X) :- disabled_person(Y), lives_with(Y,X),
                  supported_by(Y,X).

parent(Y,X)    :- father(Y,X).
parent(Y,X)    :- mother(Y,X).
```

4.4 Applying EBL to line 324

In order to apply EBL on the example, we have to specify all components of the EBL process used here. First, the non-operational goal concept is

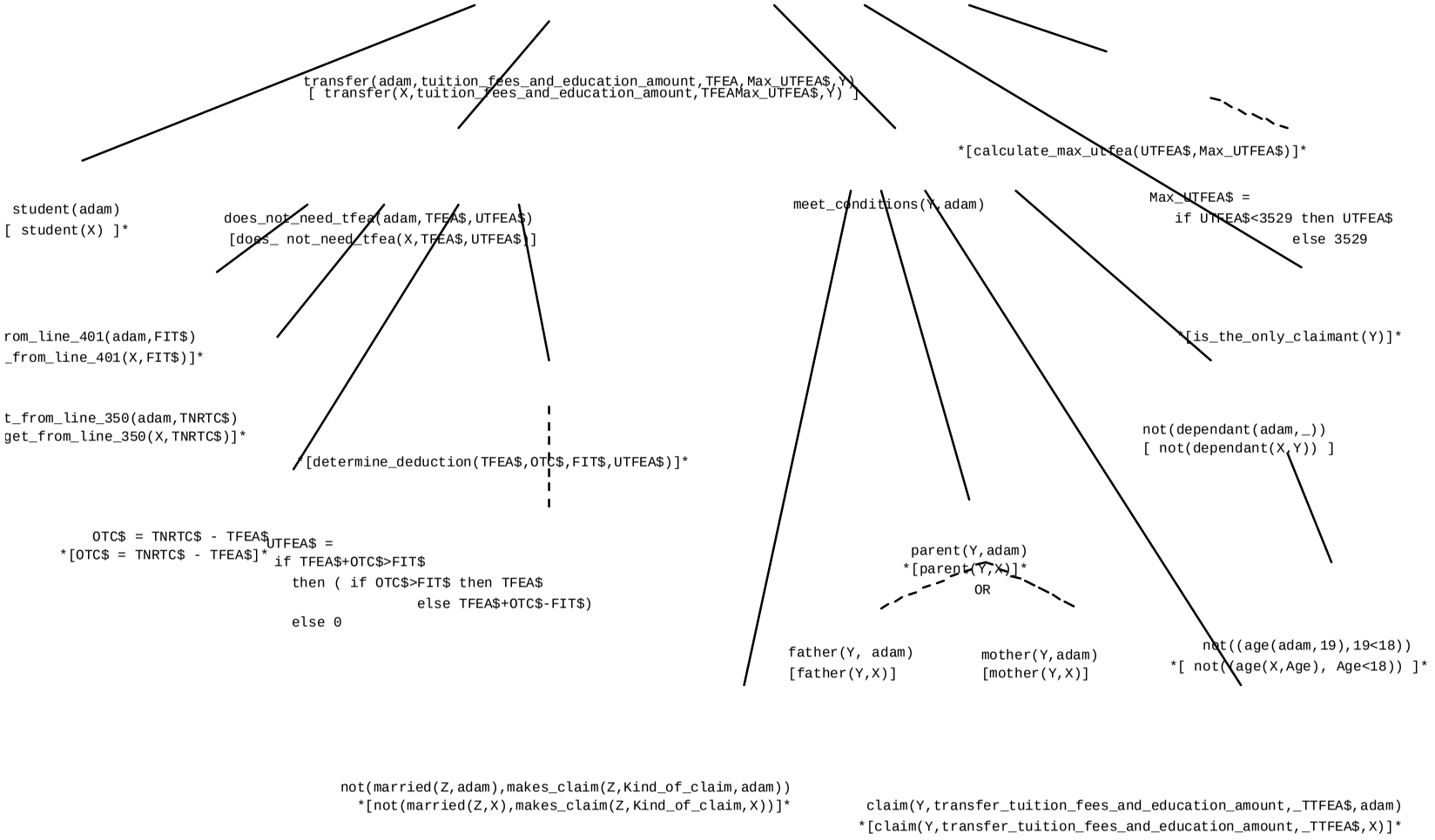
```
apply_line_324(TaxPayer, TFEA$, Max_UTFEA$, Claimant) :-
      transfer(TaxPayer, unused_part_tuition_fees_and_education_amount,
                  TFEA$, Max_UTFEA$, Claimant)
```

To satisfy the operationality criterion means to express the above goal concept in such a manner that it allows people to determine if they can transfer their ‘tuition fees and education amount’ to a parent or grandparent (or in-laws) without having to access additional knowledge or to perform unspecified computation.

Running EBL on the training example produces the explanation shown in Fig. 2. Let us observe that the tree is an AND/OR tree: its instance will depend on the actual value of the variable (TFEA$), not instantiated in the example.

The result of EBL is obtained by collecting the “leaves” (i.e., operational nodes) of the deduction tree shown in Fig. 2. These give us an operational procedure for transferring the unused part of tuition fees and education amount :

```
student(TaxPayer),
not((age(TaxPayer,Age), Age < 18)),
parent(Claimant, TaxPayer),
claim(Claimant, transfer_tuition_fees_and_education_amount,
      _, TaxPayer),
not(married(Z,TaxPayer), makes_claim(Z, Kind_of_claim, TaxPayer)),
get_from_line_350(TaxPayer, TNRTC$),
get_from_line_401(TaxPayer, FIT$),
OTC$ = TNRTC$ - TFEA$,
UTFEA$ =
   if TFEA$ + OTC$ > FIT$
         then ( if OTC$ > FIT$   then TFEA$
                                 else TFEA$ + OTC$ - FIT$)
         else 0 ,
Max_UTFEA$ =
   if UTFEA$ < 3529  then UTFEA$
                     else 3529 ,
is_the_only_claimant(Claimant).
-->
   transfer(TaxPayer,
         unused_part_tuition_fees_and_education_amount,  TFEA$,
         Max_UTFEA$, Claimant)
```


transfer(adam,tuition_fees_and_education_amount,TFEA,Max_UTFEA$,Y)
[transfer(X,tuition_fees_and_education_amount,TFEAMax_UTFEA$,Y)]
[calculate_max_utfea(UTFEA$,Max_UTFEA$)]
student(adam)
[student(X)]*
does_not_need_tfea(adam,TFEA$,UTFEA$)
[does_ not_need_tfea(X,TFEA$,UTFEA$)]
meet_conditions(Y,adam)
Max_UTFEA$ =
if UTFEA$<3529 then UTFEA$
else 3529
rom_line_401(adam,FIT$)
_from_line_401(X,FIT$)]*
[is_the_only_claimant(Y)]
t_from_line_350(adam,TNRTC$)
get_from_line_350(X,TNRTC$)]*
not(dependant(adam,_))
[not(dependant(X,Y))]
[determine_deduction(TFEA$,OTC$,FIT$,UTFEA$)]
OTC$ = TNRTC$ - TFEA$
[OTC$ = TNRTC$ - TFEA$]
UTFEA$ =
if TFEA$+OTC$>FIT$
then (if OTC$>FIT$ then TFEA$
else TFEA$+OTC$-FIT$)
else 0
parent(Y,adam)
[parent(Y,X)]
OR
father(Y, adam)
[father(Y,X)]
mother(Y,adam)
[mother(Y,X)]
not((age(adam,19),19<18))
[not((age(X,Age), Age<18))]
not(married(Z,adam),makes_claim(Z,Kind_of_claim,adam))
[not(married(Z,X),makes_claim(Z,Kind_of_claim,X))]
claim(Y,transfer_tuition_fees_and_education_amount,_TTFEA$,adam)
[claim(Y,transfer_tuition_fees_and_education_amount,_TTFEA$,X)]

The learned rule is an effective, operational procedure. Any reader of the tax guide may assess the applicability of the rule to their case, and use it to compute the appropriate amounts of credits. Moreover, no such rule was available in the domain theory. Integration of the domain knowledge, background knowledge, and the example has produced a useful, operational rule that can supplement the knowledge obtained through the text analysis process.

5. Conclusion

In this paper, we have described an implemented system that performs EBG on the output generated by a text analysis system. We have shown that such a realistic and natural application of EBG posits certain non-trivial problems necessitating extension of EBG. One of these problems is the level of operationality which can result from EBL, which varies depending the expliciteness of the training example. If all the data characterizing the training example are explicitly given, a level of operationality that does not necessitate any further evaluation beyond substitution of constants for variables can be achieved. If such a level of expliciteness is not achieved, we obtain a slightly less operational rule that needs domain-independent background knowledge in order to be applied.

We have also demonstrated that our approach can produce *useful* rules that were absent from the domain theory. Moreover, these rules generalize and compile knowledge necessary to perform a task, e.g., filling out a section of an Income Tax return. We show how to obtain such rules directly from the source document, without the need to be given a domain theory and a training example in an artificial (from the problem's point of view) representation.

6. Acknowledgements

The work described here has been performed at the laboratory of the Ottawa Machine Learning Group (OMLG). Activities of OMLG are supported by Natural Sciences and Engineering Research Council of Canada, the Government of Ontario (URIF Program), Cognos Inc, and the Canada Centre for Remote Sensing. The authors thank Dr. Stan Szpakowicz for his comments on an earlier version of the paper, and Michael MacKay for his editorial assistance.

7. References

Austin (1990) - Proc. of the 7th International Conference on Machine Learning.

Bruce, B. (1975), “Case Systems for Natural Language”, Artificial Intelligence, 6, 327-360.

Cohen, W.W. (1990), “Learning from Textbook Knowledge: A Case Study”, Proc. of AAAI-90, 743-748.

Copeck, T., Delisle, S. & Szpakowicz, S. (1990), “Intelligent Case Analysis in the KATE System”, TR-90-24, Department of Computer Science, University of Ottawa.

DeJong, G.F. & Mooney, R. (1986), “Explanation-Based Learning: An Alternative View”, Machine Learning, 1, 145-176.

Delisle, S. (1987), “A Natural Language Interface for an Expert Advisor System”, M.C.S. Thesis, Department of Computer Science, University of Ottawa.

Delisle, S. (1990), “A Parser for Processing Technical Texts with a Large Coverage of English”, TR-90-25, Department of Computer Science, University of Ottawa.

Dietterich, T.G. (1989), “Machine Learning”, Annual Review of Computer Science, 4, 1989-1990, 255-306.

Ellman, T. (1989), “Explanation-Based Learning: A Survey of Programs and Perspectives”, ACM Computing Surveys, 21, June 1989.

Fong, S. & Berwick, R.C. (1985), “New Approaches to Parsing Conjunctions Using Prolog”, in Proc. of the 23rd Annual Meeting of the ACL, 118-126.

Frey, W., Reyle, U. & Rohrer, C. (1983), “Automatic Construction of a Knowledge Base by Analyzing Texts in Natural Language”, in Proc. of IJCAI-83, 727-729.

Friedman, C. (1986), “Automatic Structures of Sublanguage Information”, in Analyzing Language in Restricted Domains: Sublanguage Description and Processing (edited by R. Grishman and R. Kittredge), LEA, 85-102.

General Tax Guide and Return - Residents of Quebec, 1989, Revenue Canada (Taxation).

Gomez, F. (1989), “Knowledge Acquisition from Natural Language for Expert Systems Based on Classification Problem-Solving Methods”, Proc. of the 4th Knowledge Acquisition for Knowledge-Based Systems Workshop.

Kedar-Cabelli, S.T. & McCarty, L.T. (1987), “Explanation-Based Generalization as Resolution Theorem Proving”, Proc. of the 4th International Workshop on Machine Learning, 383-389

Lenat, D., Guha, R.V., Pittman, K., Pratt, D. & Shepherd, M. (1990), “CYC: Toward Programs with Common Sense”, CACM, 33(8), 30-49.

Mitchell, T., Keller, R.M. & Kedar-Cabelli (1986), S.T., “Explanation-Based Generalization : A Unifying View”, Machine Learning, 1, 47-80.

Moulin, B. & Rousseau, D. (1990), “A Knowledge Acquisition System for Analyzing Prescriptive Texts”, Proc. of the 5th AAAI-Sponsored Knowledge Acquisition for Knowledge-Based Systems Workshop, 22.1-22.20.

Nishida, F., Takamatsu, S., Tani, T. & Kusaka, H. (1986), “Text Analysis and Knowledge Extraction”, Proc. of COLING-86, 241-243.

Regoczei, S. & Hirst, G. (1989), “Sortal Analysis with SORTAL, a software assistant for Knowledge Acquisition”, Proc. of the 4th Knowledge Acquisition for Knowledge-Based Systems Workshop.

Reimer, U. (1990), “Automatic Knowledge Acquisition from Texts: Learning Terminological Knowledge via Text Understanding and Inductive Generalization”, Proc. of the 5th AAAI-Sponsored Knowledge Acquisition for Knowledge-Based Systems Workshop, 27.1-27.16.

Rinaldo, F.J. (1989), “Deriving Rules for Medical Expert Systems Using Natural Language Parsing and Discourse Analysis”, Ph.D. Thesis, Computer Science Department, Illinois Institute of Technology.

Sager, N. (1981), Natural Language Information Processing: A Computer Grammar of English and its Applications, Addison-Wesley.

Salembier, M., Matwin S. & d’Alche F. (1990), “Explanation-Based Learning of Disjunctive Concepts with Non-Horn Clauses”, ISMIS ‘90, To appear.

Silvestro, K. (1988), “Using Explanations for Knowledge-Base Acquisition”, Int. J. Man-Machine Studies 29, 159-169.

Somers, H.L. (1987), Valency and Case in Computational Linguistics, Edinburgh University Press.

Szpakowicz, S. (1990), “Semi-Automatic Acquisition of Conceptual Structures from Technical Texts”, Int. J. Man-Machine Studies, 33, 385-397.