Least Generalization under Relative Implication

Svetla Boytcheva

Department of Information Technologies,
Faculty of Mathematics and Informatics,
Sofia University "St. Kliment Ohridski",
5 J. Bauchier Blvd.,
1164 Sofia, Bulgaria,
svetla@fmi.uni-sofia.bg

Abstract. The main operators in Inductive Logic Programming (ILP) are specialization and generalization. In ILP, the three most important generality orders are subsumption, implication and implication relative to background knowledge. The present paper discusses the existence of least generalization under implication relative to background knowledge. It has been shown that the least generalization under relative implication does not exists in the general case, but, as argued in this paper, it exists if the sets to be generalized and the background knowledge satisfy some special conditions.

1 Introduction

Inductive Logic Programming (ILP) is a subfield of Logic Programming and Machine Learning that investigates the problem of inducing clausal theories from given sets of positive and negative examples. An inductively inferred theory must imply all of the positive examples and none of the negative examples. The paper is organized as follows. In section 2 some preliminary definitions of the concepts used in the further discussion will be given. In Section 3 we will discuss existence of least generalization under relative implication and it will be shown that it does not exist in the general case, but exists if the sets to be generalized and the background knowledge (BK) are of some special kind. The most interesting and useful case is to find least generalization under relative implication(LGRI) (which is a set of definite program clauses) for the BK and sets of positive and negative examples that are definite program clauses. In section 3 it will be shown that in this case, after imposing some additional restrictions to the given sets, LGRI exists.

The LGRI exists for many other more particular cases of the given sets (for details see [5, 6]). However in most of them the background knowledge is a set of ground clauses or literals. Even the subsumption is weaker than implication, LGRS not exists in the general case both for the clausal language and for a Horn language. LGRS exists only for background knowledge sets of ground atoms.

2 Preliminaries

The definitions of the concepts used in the further discussion are given in this section.

Definition 1: Let Σ be a set of formulas and ϕ a formula. Then ϕ is said to be a *logical consequence* of Σ (written as $\Sigma \models \phi$), if every model of Σ is a model of ϕ. If $\Sigma \models \phi$, we also sometimes say that Σ *logically implies* (or just *implies*) ϕ. If $\Sigma = \{\psi\}$, this can be written as $\psi \models \psi$.

Definition 2: Let Σ and Γ be sets of formulas. Γ is said to be a *logical concequence* of Σ(written as $\Sigma \models \Gamma$), if $\Sigma \models \phi$, for every formula $\phi \in \Gamma$. We alseo sometimes say that Σ (*logically*) *implies* Γ.

Definition 3: Let Γ be a set and R be a binary relation on Γ.

1. R is *refflexive* on Γ if xRx for every $x \in \Gamma$.
2. R is *transitive* on Γ if for every $x, y, z \in \Gamma$, xRy and yRz impies xRz.
3. R is *symmetric* on Γ if for every $x, y \in \Gamma$, xRy impies xRy.
4. R is *anti-symmetric* on Γ if for every $x, y \in \Gamma$, xRy and xRy, implies $x = y$.

If R is both reflexive and transitive on Γ we say R is a *quasi-order* on Γ. If R is both transitive and anty-symmetric on Γ we say R is a *partial order* on Γ. If R is reflexive, transitive and symmetric on Γ we say R is a *equivalence relation*.

Definition 4: Let Γ be a set of clauses, $\geq$ be a quasi-order on Γ, $S \subseteq \Gamma$ be a finite set of clauses and $C \in \Gamma$. If $C \geq D$ for every $D \in S$, then we say that C is a *generalization* of S under $\geq$. Such a C is called a *least generalization (LG) of S under $\geq$ in Γ* if we have $C' \geq C$ for every generalization $C' \in \Gamma$ of S under $\geq$.

Dually, C is a *specialization* of S under $\geq$, if $D \geq C$ for every $D \in S$. Such a C is called a *greatest specialization (GS) of S under $\geq$ in Γ* if we have $C \geq C'$ for every specialization $C \in \Gamma$ of S under $\geq$.

Theorem 1 (Deduction Theorem): Let Σ be a set of formulas and ϕ and ψ be formulas. Then $\Sigma \cup \{\psi\} \models \phi$ iff $\Sigma \models (\psi \rightarrow \phi)$.

Preposition 1: Let Σ be a set of formulas and *phi* be a formula. Then $\Sigma \models \phi$ iff $\Sigma \cup \{\neg\phi\}$ is unsatisfiable.

Definition 5: Let B be background knowledge (set of clauses) and C and D be clauses. We will say that C *logically implies D relative to* B if $\{C\} \cup \mathrm{B} \models \mathrm{D}$ and we denote as $C \models_{\mathrm{B}} D$.

Definition 6 (Concept learning problem): Given background knowledge B and given sets of positive and negative examples P and N, the induction task of a concept learning problem is to find a concept description in the form of a logic program T that satisfies the following conditions:

1. T $\cup$ B $\models$ A for all $A \in P$ (posterior sufficiency)
2. T $\cup$ B$\neg$ $\models$ A for all $A \in N$ (posterior satisfiability)
3. B$\neg$ $\models$ A for all $A \in N$ (prior satisfiability)
4. B$\neg$ $\models$ A for all $A \in P$ (prior necessity)

Every such program T is called a *target program.*

Definition 7: Let H and B be sets of clauses and D be a clause. H is a *least generalization of D under relative implication(LGRI) to background knowledge B*, if $H \models_B D$ and for each set of clauses C, such that $C \models_B D$ is valid $C \models_B H$.

Definition 8: Let C and D be a clauses and Σ be a set of clauses. We say that C *subsumes* D, denote $C \geq D$ if there exists a substitution θ such that $C\theta \subseteq D$.

Definition 9: Let L be a first-order language. The *Herbrand universe* U_L for L is the set of all ground terms, which can be formed out of the constants and function symbols appearing in L. In case L does not contain any constants, we add one arbitrary constant to the alphabet to be able to form ground terms.

Definition 10: Let L be a first-order language. The *Herbrand base* B_L for L is the set of all ground atoms, which can be formed out of the predicate symbols in L and the terms in the Herbrand universe U_L.

Definition 11: Let L be a first-order language. The *Herbrand preinterpretation* for L is the pre-interpretation J consisting of the following:

1. The domain of the pre-interpretation is the Herbrand universe U_L.
2. Constants in L are assigned to themselves in U_L. $J(a) = a$, a-constant
3. Each n-arity function symbol f in L is assigned the mapping J_f from U_L^n to U_L, defined by $J_f(t1, \ldots, tn) = f(t1, \ldots, tn)$.

Definition 12: Let L be a first-order language and J a Herbrand preinterpretation. Any interpretation I, such that $J \subseteq I$ is called a *Herbrand interpretation.*

Definition 13: Let L be a first-order language, Σ a set of formulas of L, and I a Herbrand interpretation of L. If I is a model of Σ, it is called a *Herbrand model* of Σ.

Definition 14: Clause C *subsumes* (or is more general than) clause D with respect to logic program P if for any Herbrand interpretation I (for the language of at least P, C, D) such that P is true in I, and for any atom A, C covers A in I whenever D covers A. This is denoted $C \geq_P D$. C is referred to as a *generalization* of D, and D as a *specialization* of C.

All other concepts used above have the standard definitions. For more details see [1–3, 5, 6].

3 Existence of Least Generalization Under Relative Implication

In this section we will discuss the existence of least generalization under relative implication.

For general clauses, the LGRI-question has a negative answer. We will sketch the counter example given in [5, 6].

3.1 Example for Non-existence of LGRI in the General Case

Example 1: Even if S and the background knowledge Σ are both finite sets of function-free clauses, a LGRI of S relative to Σ does not necessarily exist. Let $D_1 = P(a)$, $D_2 = P(b)$, $S = \{D_1, D_2\}$ and $\Sigma = \{(P(a) \vee \neg Q(x)), (P(b) \vee \neg Q(x))\}$. We will show that S has no LGRI relative to Σ.

Suppose C is a LGRI of S relative to Σ. Note that if C contains the literal $P(a)$, then the Herbrand interpretation that makes P(a) true and which makes all other ground literals false would be a model of $\Sigma \cup \{C\}$ but not of D_2, so we have $C \not\models_\Sigma D_2$. Similarly if C contains $P(b)$ then $C \not\models_\Sigma D_1$. Hence C cannot contain $P(a)$ or $P(b)$.

Now let d be a constant not appearing in C. Let $D = P(x) \vee Q(d)$. Then $D \models_\Sigma S$. By the definition of the LGRI, we should have $D \models_\Sigma C$. Then by Subsumption Theorem [5], there must be a derivation from $\Sigma \cup \{D\}$ of a clause E, which subsumes C. The set of all clauses which can be derived (in 0 or more resolution-steps) from $\Sigma \cup \{D\}$ is $\Sigma \cup \{D\} \cup \{(P(a) \vee P(x)), (P(b) \vee P(x))\}$ but none of these clauses subsumes C, because C does not contain the constant d or the literals $P(a)$ and $P(b)$. Hence $D \not\models_\Sigma C$ contradicts the assumption that C is a LGRI of S relative to Σ.

Thus, in general LGRI of S relative to Σ need not exist.

3.2 Analyses of Some Properties of the Given Sets

Where is the weak point? Let's look again on the background knowledge set $\Sigma = \{(P(a) \vee \neg Q(x)), (P(b) \vee \neg Q(x))\}$. We can present this set in the following equivalent form $\Sigma = \{(Q(x) \rightarrow P(a)), (Q(x) \rightarrow P(b))\}$. The BK set Σ consists of Horn clauses and we can represent it as the program:

```
p(a):-q(X).
p(b):-q(X).
```

We can see that two different ground instances ($P(a)$ and $P(b)$) of the predicate $P(x)$ can be inferred from an arbitrary grounding of $Q(x)$ inferences. One of the possible generalizations of the given set relative to the BK is:

```
p(Y):-q(X).
```

But there is no dependency between the variables X and Y, and this is not a useful generalization, because it is not generative. Thus, some restrictions on the BK and the set to be generalized must be made to ensure the existence of a LGRI. Some examples of cases when a LGRI does exist will be given and after analysing them we will formulate the requirements for the BK and the initial set. Let the BK $\Sigma = \{C_1, C_2, \ldots, C_m\}$ be a finite set of clauses and $S = \{D_1, D_2, \ldots, D_n\}$ be a finite set of clauses. Additionally we suppose that:

- a substitution θ, such that $C_{ibody}\theta = C_{jbody}$, for $i \neq j$ does not exist
- a predicate A such that $A' \in C_{ihead}$ and $A'' \in C_{jhead}$, where A' and A'' are ground instances of A, does not exist.

Example 2: Consider the following set of positive examples:

```
C1 = food(X):-tasty(X),strawberry(X).
C2 = food(X):-tasty(X), not_poisonous(X), mushroom(X).
```

The most obvious way to generalize them is to take their least generalization under implication, which is the rather general and not very useful clause:

```
D = food(X):-tasty(X).
```

Suppose we have the following definite program $\Sigma = \{B_1, B_2, B_3\}$, expressing background knowledge:

```
B1 = plant(X):-mushroom(X).
B2 = plant(X):-strawberry(X).
B3 = not_poisonous(X):-strawberry(X).
```

Taking Σ into account, we may also find the more informative generalization clause:

```
D' = food(X):- tasty(X), not_poisonous(X), plant(X).
```

D' together with Σ implies both examples, but without the BK our clause D' does not imply the examples. For instance, not everything that has delicious taste is eatable, some things can be poisonous or harmful for people.

In their article [7], Muggleton and Buntine described two operators based on inverting resolution steps: the V- and the W-operator (fig.1).

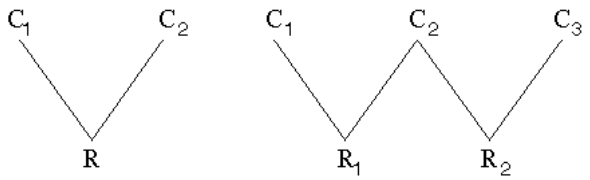

Fig.1. The main view of the V-operator and the W-operator

Given C_1 and R, the V-operator finds C_2 such that R is an instance of a resolvent of C_1 and C_2. Thus the V-operator generalizes $\{C_1, R\}$ to $\{C_1, C_2\}$. The W-operator combines two V-operators, and generalizes $\{R_1, R_2\}$ to $\{C_1, C_2, C_3\}$, such that R_1 is an instance of a resolvent of C_1 and C_2, and R_2 is an instance of a resolvent of C_3 and C_2. In addition the W-operator is able to invent new predicates.

Going back to the example described above it is easy to see, that D' is a result of consecutively applying V- (see Fig.3) and W-operators(see Fig.2) under C_1 , C_2 and clauses of Σ.

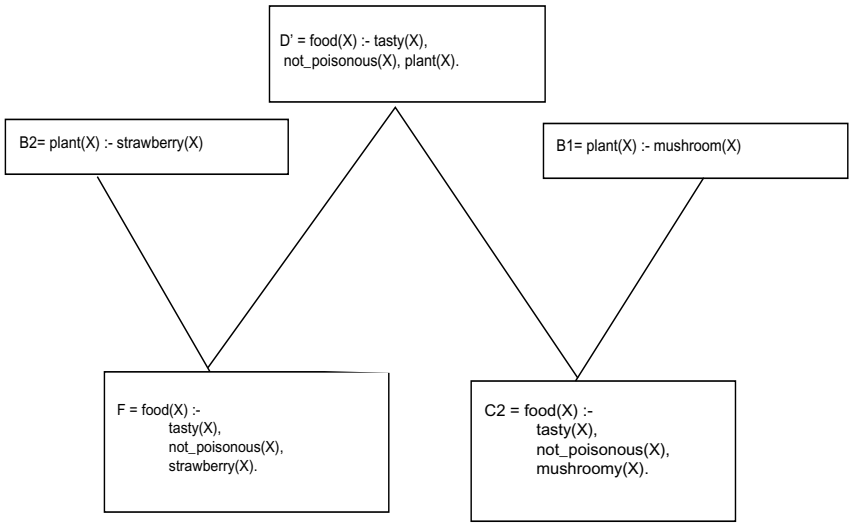

Fig.2. The W-operator applyed on C_2, B_1, B_2 and F

Let D is the result of the W-operator applyed on C_2, B_1, B_2 and F. D is the LGRI of $\{C_1, C_2\}$ under $\{B_1, B_2, B_3\}$.

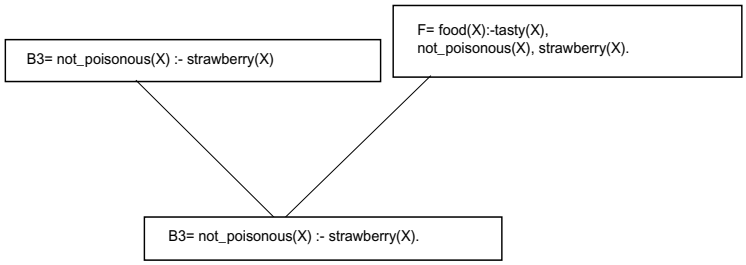

Fig.3. The V-operator applied on C_1 and B_3

Let F_1 is the result of the V-operator applied on C_1 and B_3.

3.3 More Definitions

These two operators require some restrictions on the type of the given clauses. The following definitions will help us to describe some of them.

Definition 15: Let C be a clause. C is a *generative clause* if all variables in C_{head} are contained in C_{body}.

Definition 16: Let C be a clause and Σ be a set of clauses. Let C contain n different variables. C is a *determined clause* with respect to Σ if after binding $n-1$ variables of C with terms of Σ for the remining variable of C there exists a unique substitution that binds this variable with a term contained in Σ.

Definition 17: Let $\Sigma = \{C_1, C_2, \ldots, C_m\}$ and $S = \{D_1, D_2, \ldots, D_n\}$ be finite sets of clauses. S has an *absolute model* under Σ if for each $D_i \in S$ and for each literal $L \in D_i body$ there exists a clause $E = \{some\ C_j \in \Sigma\ or\ some\ D_j \in S\}$, and a substitution σ such that $L\sigma \in E$.

For a clause C_2 to exist, the V-operator requires C_1 and R to be generative clauses.

For clauses C_1, C_2 and C_3 to exist, the W-operator requires R_1, R_2 to be generative clauses. The found clause C_3 is generative and determined with respect to the set $\{R_1, R_2, C_1, C_2\}$.

The clauses R_1, R_2 have one and the same head, hence the clause C_3 will have the same head and the clause C_2 will be a generalization of the set of clauses $\{R_1, R_2, C_1, C_3\}$.

Suppose that R_1, R_2 are members of the given set of clauses, that we would like to generalize and C_1 and C_3 are clauses from the background knowledge.

We can consider C_2 as a generalization under implication of R_1, R_2 relative to background knowledge set $\{C_1, C_3\}$.

The clause C_2 is a LGRI of R_1, R_2 because it is generated by one resolution step.

If the set that will be generalized has an absolute model under background knowledge then we can easily combine clauses from the given set and the BK in V- and W-operators.

The previous discussion enables the formulation of the following theorem.

3.4 Theorem of Existence of LGRI in Limited Case

Theorem 2: *Let $\Sigma = \{C_1, C_2, \ldots, C_m\}$ be a finite set of function-free definite program clauses and $S = \{D_1, D_2, \ldots, D_n\}$ be a set of function-free definite program clauses and all D_i have the same predicate symbol in their heads and at least one of them is non-tautologous. If S has an absolute model under Σ and all the clauses of S are generative and $\Sigma \not\models S$, then a LGRI of S relative to Σ exists.*

Proof: Let $\Sigma = \{C_1, C_2, \ldots, C_m\}$ be a finite set of function-free definite program clauses and $S = \{D_1, D_2, \ldots, D_n\}$ be a set of function-free definite program clauses and all D_i have the same predicate symbol in their heads and at least one of them is non-tautologous. The LGRI T of S relative to Σ exists if for every $C_i, C_j \in \Sigma$ there does not exist a substitution θ such that $C_{ibody}\theta = C_{jbody}$, and there does not exist a predicate A such that $A' \in C_{ihead}$ and $A'' \in C_{jhead}$, where A' and A'' are ground instances of A and for every $D_i \in S$ and for every literal $L \in D_i$ there exists clause $E = \{some\ C_j \in \Sigma\ or\ some\ D_j \in S\}$ and a substitution σ such that $L\sigma \in E$.

Then $T \not\models_\Sigma D_i$ iff $\{T\} \cup \Sigma \models D_i$ iff $T \models D_i \cup \neg\Sigma$. It remains to be shown that $D_i \cup \neg\Sigma$ is a set of function-free clauses and at least one of them is non-tautologous. Then by the theorem for existence of the least generalization under implication (LGI) [5, 6], it will follow that a generalization H exists. The clauses of the set $D_i \cup \neg\Sigma$ are function-free, as required in the condition of the theorem. Since each $D_j \in S$ has the same predicate in its head, each clause in $T = \{(D_1 \cup \neg\Sigma), (D_2 \cup \neg\Sigma), (D_n \cup \neg\Sigma)\}$ will contain the same predicate in its head.

Because of the conditions of the theorem, each of the elements of T is a definite program clause.

It remains to show that $T = \{(D_1 \cup \neg\Sigma), (D_2 \cup \neg\Sigma), (D_n \cup \neg\Sigma)\}$ contains at least one non-tautologous clause. Suppose that all clauses in T are tautologous. From the definition of tautologous clause we conclude that every interpretation is a model of the clauses in T, in other words $\models D_i \cup \neg\Sigma$, hence $\Sigma \models D_i$ for each $D_i \in S$, hence $\Sigma \models S$, but this is a contradiction with the theorem conditions.

So, T is a set of definite program clauses and at least one of them is non-tautologous. From the theorem for existence of LGI (see [5, 6]), we obtain that there exists a LGI H of T and H will be a LGRI of S relative to Σ.

Why do we need the sets' restrictions in the theorem 2? Are they too strong or not?

Most of the restrictions are necessary, because of the V- and W-operators requirements for the existence of the generalization clause and its computability.

The restriction $\Sigma \not\models S$ is imposed by the definition of the concept learning problem (prior necessity).

The restriction of the set S to contain one and the same predicate symbol in their heads is imposed by the necessity for the obtained LGRI of S under BK to be a program, that gives the definition of the concept, coded by this predicate symbol.

The other restrictions come from the analysis of the contradiction example mentioned above and from the requirement for the background knowledge to be consistent.

3.5 Computability of LGRI

This LGRI is computable because, it is a kind of LGI, which is computable. There are exists algorithm for construction of LGI of given sets. This algorithm is not very efficient. A more efficient algorithn may exist but since implication is harder than subsumption and the computation of an LGS is already quite expensive we should not put our hopes too hight. Nevertheless the existence of the LGI-algorithm does establishe the theoretical point that the LGI of any finite set of clauses containing at least one non-tautologous function-free clause is effectively computable.

4 Conclusion

The presented case of existence of a least generalization under relative implication helps us to search for generalizations of the concepts presented by most natural and often used types of sets and background knowledge. In the concept learning problem, usually examples are presented as ground literals and/or definite program clauses, and the background knowledge is a program. It is reasonable to expect that the LGRI of these sets will be a program too.

The contribution of this paper is the discovery of a more general (than those described in the literature) case of existence of least generalization under relative implication. This result can be used for several applications in the field of Machine learning, such as automated generation of concept definitions, improvement of predicate definitions and other kinds of concept generalization.

In the further work a simpler algorithm for finding the least generalization under relative implication will be presented concerning the described cases.

Another line of research is to find other cases of existence of LGRI.

5 Acknowledgements

I would like to thank Zdravko Markov and Ani Nenkova for their comments, which helped to improve this paper.

References

1. W.Buntine, Generalized Subsumption and Its Applications to Induction and Redundancy, Artificial Intelligence, 36(2),149-176, (1988).
2. P. van der Laag, An anlyses of refienment operators in inductive logic programming, Ph.D thesis, Tinberg Institute Research Series, Rotherdam, (1995).
3. J.W.Lloyd, Foundations of Logic Programming, Springer-Verlag Berlin Heidelberg, (1984).
4. Z. Markov, Machine Learning, Softex, (1996) (in Bulgarian).
5. S-H. Nienhuys-Cheng and R. de Wolf, Foundations of Inductive Logic Programming, Springer-Verlag,Berlin Heidelberg, (1997).
6. S-H. Nienhuys-Cheng and R. de Wolf, Least generalizations and greatest specializations of sets of clauses, Journal of Artificial Intelligence, 4:341-363, (1996).
7. S. Muggleton and W. Buntine, Machine invention of first-order predicates by inverting resolution. In J.Laird editior, Proceedings of the 5th International Conference on Machine learning (ICML-88),pages 339-352, Morgan Kaufman, San Mateo, CA, (1988).