The Relation between Abductive Hypotheses and Inductive Hypotheses

Akinori Abe
NTT Communication Science Laboratories
1-1 Hikarino-oka Yokosuka-shi Kanagawa 239 JAPAN
ave@cslab.kecl.ntt.co.jp

Abstract

In the AI field, abduction is used for design, diagnosis and other tasks. This is because abduction is a sort of predictive inference. On the other hand, induction is used for classification. For example, ILP finds greater tendencies in examples and learns general rules from the examples and background knowledge. Recently, the integration of abduction and induction has become a hot issue. In this paper, first, I show the relationship between abductive hypotheses and inductive hypotheses, then I present a solution for integration of abduction and induction.

1 Introduction — the relationship between abduction and induction

1.1 Abduction and induction

Peirce wrote that abduction is an operation for adopting an explanatory hypothesis, which is subject to certain conditions, and that in pure abduction, there can never be justification for accepting the hypothesis, other than through interrogation [Peirce, 1955]. In AI, if we make use of the framework of logic, a sort of justification will be made by generating and testing candidate hypotheses to find acceptable hypotheses. Therefore, in simple words, abduction can be formalized as an explanation of an observation. It generates or adopts consistent hypotheses to explain an observation. Furthermore, abduction can be formalized as prediction because hypotheses generation and testing can be seen as a prediction of the causation of an observation. Peirce also wrote that induction is an operation for testing a hypothesis by experiment, which consists in remarking that, if it is true, an observation made under certain conditions ought to have certain results. Then, causing those conditions to be fulfilled and noting the results, if they are favourable, we extend a certain confidence to the hypothesis. Thus, simply speaking, induction can be formalized as the generalization of examples. It finds tendencies in examples, to generate general rules (hypotheses) from examples and background knowledge.

Both inferences seem to be the same because both can be viewed as being the inverse of deduction. However, the main difference between abductive hypotheses and inductive hypotheses is that, in general, abductive hypotheses are propositional clauses and inductive hypotheses are predicate clauses. Intuitively, this difference comes from the method of inference: Abduction involves the adoption of existing or non-existing hypotheses for an explanation of a given observation. Therefore, the hypotheses are usually propositional clauses. On the other hand, induction involves the classification or generalization of examples for explaining tendencies in observations. Therefore, the hypotheses become predicate clauses.

1.2 Abduction

Theorist

One of the popular systems of abduction is Theorist [Poole, Aleliunas and Goebel, 1987], which is a hypothetical reasoning system. The inference mechanism of Theorist is to explain an observation (O) by consistent hypotheses (h). Hypotheses are selected from the set of hypotheses.

$$F \not\vdash O. \tag{1}$$

$$F \cup h \vdash O. \tag{2}$$

$$F \cup h \not\vdash \Box. \tag{3}$$

where F is a fact (always true and can not have any inconsistency with other knowledge), h is a hypothesis (defeasible knowledge which requires consistency checking during the inference process) O is the observation to be explained, and $\Box$ is an empty clause.

Theorist can not generate new hypotheses, it only adopts consistent hypotheses (usually atom-type clauses) from the hypotheses set.

CMS

Another popular abduction system is CMS [Reiter and de Kleer, 1987]. The inference mechanism of CMS is as follows: When $\Sigma \not\models C$, if propositional clause C (observation) is given, CMS returns a set of minimal clauses S to clause set Σ such that

$$\Sigma \models S \vee C. \tag{4}$$

$$\Sigma \not\models S. \tag{5}$$

A clause S is called a minimal support clause, and $\neg S$ is a clause missing from clause set Σ that can explain C. Therefore, from the abductive point of view, $\neg S$ can be thought of as an abductive hypothesis. This hypothesis is not included in the knowledge base ($\Sigma \not\models S$.), and is not necessarily to be an atom-type clause.

1.3 Induction

In the realm of induction, ILP is now one of the hottest research areas, and various ILP systems have been proposed over the past ten years. ILP systems may receive various kinds of information about the desired program as input, but this input always includes examples of the program's input/output behavior. The output that is produced is a logic program that behaves as expected on the given examples, or at least on a high percentage of them [Bergadano and Gunetti, 1996]. The mechanism of ILP is as follows:

$$E = E^+ + E^- \tag{6}$$

$$B \cup h \models E^+. \tag{7}$$

$$B \cup h \cup E^- \not\models \Box. \tag{8}$$

where B is background knowledge and h represents hypotheses. E denotes examples, and E can be separated into E^+ (positive examples) and E^- (negative examples).

If a set H (hypotheses space: possible program), a set E^+, a set E^-, and consistent logic program B are given such that

$$B \not\vdash e^+, \text{ for at least one } e^+ \in E^+. \tag{9}$$

Then ILP finds a logic program $h \in H$, such that B and h is complete (7) and consistent (8).

If E^- is the empty set, formulae (2) and (3) are similar to formulae (7) and (8). That is, h in formulae (2) and (3) corresponds to E^+ in formulae (7) and (8), and h in formulae (7) and (8) corresponds to F in formulae (2) and (3). In this sense, abduction can generate propositional examples that can be used in induction in a predicative way, and induction can generate facts (background knowledge) that can be used in abduction. Therefore, if abduction is performed more than once, and various abduced hypotheses are found under the same or similar observations, and if, using these abduced hypotheses as examples, induction is performed, abduction and induction can be integrated. Thus, abduction generates examples for induction and induction generates facts for abduction.

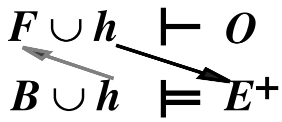

Figure 1: The relation between abductive hypotheses and inductive hypotheses.

2 Abductive Analogical Reasoning

2.1 Problems in criterion for accepting hypotheses

In Sec. 1, I mentioned two popular abduction systems. In both, the criterion for accepting hypotheses is a simplicity criterion, i.e. "Occam's Razor". However, in applications like natural language processing, using a simplicity criterion is insufficient. Such a criterion considers neither the meaning of hypotheses nor the relationships among them. Therefore, problems will occur in the process of selecting the best hypotheses. Ng [Ng and Mooney, 1990] has shown that some notion of explanatory coherence is more important in deciding which explanation is the best and introduced coherence metrics in order to select better explanations.

I have adopted the notion of explanatory coherence as analogical mapping and proposed Abductive Analogical Reasoning (AAR) [Abe, 1996].

2.2 Abductive Analogical Reasoning

In this section, I will briefly explain AAR. Before doing so, I will define some notations.

Definition 1 analogical clause: *In general, "A is an analogical clause of B" means that B is analogically mapped from A. Usually, the relation is a high-order relation like structure-mapping [Gentner, 1983], [Gentner, 1988]. However, in [Abe, 1996], I simplify analogical mapping. Let A be $a \vee b \vee \ldots \vee l$. Analogical clause of A (=B) is such a clause that has at least one atom that is similar to that of A and other atoms are the same as those of A. For example, if a is similar to a', B is $a' \vee b \vee \ldots \vee l$.*

Definition 2 *Let A and B be sets of clauses. "$A \models\!\!\approx B'$" is the relationship between A and B', iff there exists a set of clauses B' such that:*

(*i*) $A \models B$,

(*ii*) $B \neq B'$, *and*

(*iii*) *every clause in* B' *is either included in* B, *derivable from* B, *an analogical clause of that included in* B, *or an analogical clause of that derivable from* B.

Definition 3 *Let* A *be a set of clauses. "*$A \mid\!\sim\!\!\rightarrow A'$*" is the relationship between* A *and* A', *such that* A' *is a set of clauses that can derive analogical clauses of the clauses derived from* A. *Here, "*$\mid\!\sim\!\!\rightarrow$*" means "can derive clauses with analogical mapping."*

Then,

> Let Σ, A and A' be sets of clauses. When $\Sigma \models A$, if $A \mid\!\sim\!\!\rightarrow A'$, then $\Sigma \approx\!\models A'$.

AAR explains an observation, C, by satisfying the following formulae.

If observation C is given, AAR tries to explain it using clauses in the knowledge base Σ. When $\Sigma \not\models C$, Σ lacks the clauses to explain C. Then, AAR returns a set of minimal clause S such that

$$\Sigma \models S \vee C. \tag{10}$$

$$\Sigma \not\models S. \tag{11}$$

S is the clause that is necessary to explain C. However, since $\Sigma \not\models S$, justification of S is not guaranteed. Therefore, it generates clause S' such that

$$\Sigma \approx\!\models S' \vee C. \qquad (\mathrm{S} \mid\!\sim\!\!\rightarrow S') \tag{12}$$

$$\Sigma \models S'. \tag{13}$$

and generates clause S'' such that

$$\Sigma \models S'' \vee C. \qquad (S' \mid\!\sim\!\!\rightarrow S'') \tag{14}$$

$$\Sigma \not\models S''. \tag{15}$$

then it explains C by $\neg S''$ as hypotheses. S'' is logically equivalent to S.

AAR generates clauses that are necessary for explaining an observation, transforms these clauses to plausible hypotheses for explaining an observation by referring to clauses in the knowledge base, and then explains the observation by those hypotheses.

Clauses in Σ	Results from Conceptual Base
$\neg east \vee palace.$	$horse \approx_{con} donkey$
$\neg ride \vee east.$	...
$\neg palace \vee beautiful.$	
$\neg desert \vee donkey.$	
$\neg horse \vee \neg saddle \vee ride.$	
$\neg town \vee car.$	
$desert.$	
$saddle.$	

Figure 2: An example knowledge base

For example, if the knowledge base contains the clauses shown in Fig. 2, and the user gives a query "$\neg donkey \wedge palace$", the set of the minimal support clauses (S) is $\{donkey \wedge \neg palace,\ donkey \wedge \neg east,\ donkey \wedge \neg ride,\ donkey \wedge \neg horse\}$. The negation of these clauses seem to be necessary for explaining an observation. And, since "donkey" is similar to[1] "horse," by referring to '$\neg horse \vee \neg saddle \vee ride$," using analogical mapping from $\{\neg(donkey \wedge \neg ride),\ saddle\}$ in the knowledge base, the hypothesis returned from AAR is $\{\neg donkey \vee \neg saddle \vee ride,\ saddle\ \}$. This is a more plausible hypothesis than "$\neg donkey \vee ride$."

2.3 Related works

NLAG [Greiner R, 1988] is an analogical inference system. It is explicitly given analogical hints lile $A \sim B$, and then performs analogical inference. Let Th be theory, PT be a target problem, and $\varphi(A)$ a proposition, the inference is done as follows:

$$Th \not\models \varphi(A) \tag{16}$$

$$Th \not\models \neg\varphi(A) \tag{17}$$

$$Th \models \varphi(B) \tag{18}$$

$$Th \cup \{\varphi(A)\} \models PT \tag{19}$$

The philosophy of inference and the mechanism of making an analogical clause in NLAG is similar to that of AAR. However, AAR is not explicitly given any analogical hints. AAR sometimes can find the candidate target by referring to minimal support clauses. Upon doing so, it calculates the similarity between the source and candidate target, then decides if the source is similar to candidate target. In addition, how hypotheses are found in NLAG is different from AAR.

[1]The possible similarity relation can be easily found from the inference path. For details, see [Abe, 1996]

DBAR

DBAR (Determination based Analogical Reasoning) [Russell, 1988] is a solution to the justification problem and the non-redundancy problem. In DBAR, a determination relation is considered as a logical generalization of the different types of dependency relations defined in database theory and determination rules facilitate sound rule inference and valid conclusions projected by analogy from single instance, without implying what the conclusion should be prior to an inspection of the instance. The determination rule (P determines Q) is

$$P(\underline{x}, \underline{y}) \succ Q(\underline{x}, \underline{y}) \textit{ iff}$$
$$\forall \underline{w}\underline{y}\underline{z}[\ P(\underline{w}, \underline{y}) \wedge Q(\underline{w}, \underline{z}) \implies \forall \underline{x}[\ P(\underline{x}, \underline{y}) \Rightarrow Q(\underline{x}, \underline{z})\ ]\ ] \quad (20)$$

where, $\underline{x}$ is a set of variables such as $\{a, b, c, \ldots\}$.

The derived solution can be justified; however, it is unnatural to assume that one determination rule can determine every relation. Nonetheless, if the determination rule can be induced for several relations, it is useful in the analogical mapping procedure.

3 A method to integrate abduction and induction

3.1 Problems in integrating abduction and induction

Above, I showed the relationship between abduction and induction, and the difference between abductive hypotheses and inductive hypotheses. From the viewpoint formalizing abduction as prediction and induction as generalization, it is a natural and nice idea to generate propositional examples by abduction and then generate predicate rules from those abduced examples, previously existing examples and background knowledge. However, there exist some problems. In usual applications, hypotheses generated by abduction, like hypothetical reasoning systems, are thought to be positive examples. It is very rare for hypotheses to be generated when observations are negative. Therefore, in a simple integration, the induction process must be done with only positive examples. However, learning rules from only positive examples is harder than learning them from positive and negative examples; there are some restrictions (Subset Principle [Angluin, 1980], [Berwick, 1986]).

Therefore, some negative examples may be generated by abduction to perform good induction. And it is slightly easier to induce with negative examples. For example, the recent result for the ILP system, Progol [Muggleton, 1995], needed negative examples to generate rules. To overcome this problem, near miss examples are often intentionally added to an example set as negative examples. For example, AILP [Adé and Denecker, 1995], a system for integrating induction and abduction, adopts abduction to resolve the positive goal, treats a negated positive goal as a negative goal, and then generates negative examples. However, it is slightly unnatural to explain a negative observation to generate negative examples intentionally.

Esposito introduced a method for generating negative examples from integrity constraints [Esposito et al., 1996]. It seems to work well, however, in hypothetical reasoning. Usually, such no-good clauses do not explicitly appear as a result of inference.

Then, how can we properly perform induction without generating negative examples intentionally?

3.2 Solution: Collecting examples by AAR

I have adopted the notion of explanatory coherence as analogical mapping and proposed Abductive Analogical Reasoning (AAR). In AAR, analogical mapping guarantees explanatory coherence among hypotheses. It works as a constraint or a reference to generate plausible hypotheses. Since it transforms a logically correct and minimal clause to a logically correct and plausible clause by referring to known clauses, explanatory coherence is a criterion for generating plausible abductive hypotheses. Furthermore, hypotheses generated by analogical mapping from a similar source clause are classified into the same category. If AAR generates hypotheses from similar observations, hypotheses collected by AAR can become good examples for induction. However, they are all positive examples.

Furukawa and Shimizu introduced the algorithm to generate negative examples for Progol [Furukawa and Shimazu, 1996]. In this algorithm, examples in another class are considered to be negative ones of those in a certain class which are considered to be positive ones, then good negative examples are selected which are near miss examples of positive examples. In some applications like CAD, it seems to work well. However, in general, it will be too strong to divide negative examples and positive ones by such a classification.

Therefore, in this paper, I focus upon learning from positive-only examples. One of the restrictions to learning rules from positive-only examples is the Subset Principle. Let L_i be an indexed family of non-empty languages. The necessary and sufficient condition for Subset Principle is as follows:

(i) T_i is finite, where T_i is a set of strings,

(ii) $T_i \subseteq L_i$, and

(iii) for all $j > i$, if $T_i \subseteq L_i$ then L_j is not a proper subset of L_i.

i.e., for all i, hypothesis($i + 1$) is not a proper subset of hypothesis(i). Berwick writes that if hypotheses follow the Subset Principle, they could partially overlap [Berwick, 1986].

As to abduced hypotheses, the following feature can be given.

In general, a hypothesis abduced from a certain observation can never be a proper subset or superset of that abduced from a similar observation, even if they may sometimes partially overlap.

In hypothetical reasoning, if hypothesis ($\neg S$) generated from a certain observation is a subset of that ($\neg S'$) generated from a similar observation, a certain observation must be a subset of this similar observation.

This is because if we let α be a non-empty clause set and $S' = S \vee \alpha$, the following formulae hold.

$$\Sigma \models S \vee C. \tag{21}$$

$$\Sigma \models S' \vee C' = S \vee \alpha \vee C'. \tag{22}$$

Therefore,

$$C = C' \vee \alpha. \tag{23}$$

Furthermore, analogical reasoning can be formalized under the Theorist's framework as follows [Goebel, 1989]:

$$S \cup T \not\models G. \tag{24}$$

$$S \cup T \cup M \models G. \tag{25}$$

$$S \cup T \cup M \not\models \neg m. \ \ (m \in M) \tag{26}$$

where S is source knowledge, T is target knowledge, M is an equality assumption, and G is an analogical inference (axiom).

If T is a subset of S, i.e., $S = T \cup A$, where A is a non-empty clause set, then formulae (24), (25) and (26) become

$$T \cup A \not\models G. \tag{27}$$

$$T \cup A \cup M \models G. \tag{28}$$

$$T \cup A \cup M \not\models \neg m. \ \ (m \in M) \tag{29}$$

Regarding these formulae, T should be an empty set. Therefore, source knowledge cannot be a superset or subset of target knowledge. Thus, from (23), a hypothesis generated from a certain observation is not a subset or superset of that generated from a similar observation.

If hypotheses are abduced by AAR under similar observations and rules are induced from these abduced hypotheses as positive examples, abduction and induction can be integrated. The key to the integration of abduction and induction is analogy or analogical mapping. Analogical mapping from clauses in a knowledge base can guarantee explanatory coherence among hypotheses. This guarantees the generation of plausible hypotheses. And hypotheses generated under similar situations are good positive examples, so, induction of plausible rules with only positive examples can also be guaranteed.

4 Conclusion

Abductive hypotheses and inductive hypotheses are usually treated as if they were the same type of hypotheses. Indeed, the forms of the formulae for abduction and induction are similar. However, I do not think abductive hypotheses and inductive hypotheses are of the same type. I take the position that they are different, and have shown the relationship between abduction and induction. The relationship is shown in Fig. 1. It gives a clue to integrating abduction and induction. From the viewpoint of this relationship, my solution to their integration is

1) to generate propositional hypotheses by abduction under similar observations, and

2) to generate predicate rules by induction from abduced hypotheses. Then, put them into knowledge base as facts.

However, there exist some restrictions to generating a predicate rule by induction. Since the abduction procedure generates only positive examples, the induction procedure must generate rules from only positive examples. In general, it is slightly hard to learn rules from only positive examples. However, if examples satisfy the Subset Principle, new rules can be generated from only positive examples.

The solution to problem above is to generate hypotheses under similar observations. As shown in Sec. 3, if AAR generates hypotheses under the similar observations, the generated hypotheses can satisfy the Subset Principle. And inductive hypotheses (rules) can be generated from only positive examples. Furthermore, nevertheless the mapping function is presently very simple, AAR can generate plausible hypotheses by analogical mapping from clauses in the knowledge base. Therefore, induced rules are also plausible ones.

In [Abe, 1996], I introduced only the framework for generating hypotheses in abductive reasoning; the mapping function is still a simple one. If the mapping function is improved, it will be possible to produce hypotheses that are more plausible than those that can be produced at present, and the induced rules will become better ones.

Dimopoulos and Kakas suggested that abduction can help exploit high-level background theory available for learning and help to handle possible incompleteness in the background theory [Dimopoulos and Kakas, 1996]. Their concept for integration of abduction and induction is similar to mine. However, it uses negative observations to eliminate inaccurate theories. Instead of generating negative examples, it refines existing rules. Although it uses negative observations, it seems to be another way to learn with only positive examples.

References

[Abe, 1996] Abe A.: Abductive Analogical Reasoning, *proc. of the 10th. Annual Conf. of JSAI*, S6-04, pp. (123)–(126), 1996. (in Japanese)

[Adé and Denecker, 1995] Adé H. and Denecker M.: AILP: Abductive inductive Logic Programming, *proc. of IJCAI95*, pp. 1201–1207, 1995.

[Angluin, 1980] Angluin D.: Inductive Inference of Formal Languages from Positive Data, *Information and Control*, vol. 45, pp. 117–135, 1980.

[Bergadano and Gunetti, 1996] Bergadano F. and Gunetti D.: *Inductive Logic Programming*, MIT Press, 1996.

[Berwick, 1986] Berwick R. C.: Learning From Positive-only Examples: The Subset Principle and Three Case Studies, *Machine Learning*, vol. 2, pp. 625–645, Morgan Kaufman, 1986.

[Dimopoulos and Kakas, 1996] Dimopoulos Y. and Kakas A.: Abduction and Learning, *in Advances in Inductive Logic Programmings (De Raedt Eds.)*, IOS Press, 1996.

[Esposito et al., 1996] Esposito F. et al.: Learning abductive logic programs, *Workshop notes of ECAI'96 Workshop on Abductive and inductive Reasoning*, pp. 23–30, 1996.

[Furukawa and Shimazu, 1996] Furukawa K. and Shimazu K.: Knowledge Discovery in Database by PROGOL — Conception and Design, *proc. of the 10th. Annual Conf. of JSAI*, 02-03, pp. 49–52, 1996. (in Japanese)

[Gentner, 1983] Gentner D.: Structure-Mapping: A Theoretical Framework for Analogy, *COGNITIVE SCIENCE*, vol.7, pp.155–170, 1983.

[Gentner, 1988] Gentner D.: Analogical Inference and Analogical Access, *Analogica*, pp.63–88, Pitman, 1988.

[Goebel, 1989] Goebel R.: A sketch of analogy as reasoning with equality hypotheses, *Proc. of Int'l Workshop Analogical and Inductive Inference (LNAI-397)*, pp. 243–253, 1989.

[Greiner R, 1988] Greiner R.: Learning by Understanding Analogies, *Analogica*, pp.1–36, Pitman, 1988.

[Muggleton, 1995] Muggleton S.: Inverse Entailment and Progol, *New Generation Computing*, vol. 13, pp. 245–286, 1995.

[Ng and Mooney, 1990] Ng H. T. and Mooney R. J.: On the Role of Coherence in Abductive Explanation, *Proc. of AAAI90*, pp. 337–342, 1990.

[Peirce, 1955] Peirce C. S.: Abduction and Induction, chap. 11, *Philosophical Writings of Peirce*, Dover, 1955.

[Poole, Aleliunas and Goebel, 1987] Poole D., Aleliunas R. and Goebel R.: Theorist: A Logical Reasoning System for Defaults and Diagnosis, *The Knowledge Frontier: Essays in the Representation of Knowledge (N.J.Cercone, G.MaCalla Eds.)*, Springer-Verlag, 1987.

[Reiter and de Kleer, 1987] Reiter R. and de Kleer J.: Foundation of assumption-based truth maintenance systems: preliminary report, *Proc. of AAAI87*, pp.183–188, 1987.

[Russell, 1988] Russell S.J.: *The Use of Knowledge in Analogy and Induction*, Pitman, 1988.