# Identification through Inductive Verification
**Application to Monotone Quantifiers**

Nina Gierasimczuk*
Institute for Logic, Language, and Computation, University of Amsterdam
Institute of Philosophy, University of Warsaw
nina.gierasimczuk@gmail.com

## Abstract

*In this paper we are concerned with some general properties of scientific hypotheses. We investigate the relationship between the situation when the task is to verify a given hypothesis, and when a scientist has to pick a correct hypothesis from an arbitrary class of alternatives. Both these procedures are based on induction. We understand hypotheses as generalized quantifiers of types $\langle 1\rangle$ or $\langle 1,1\rangle$. Some of their formal features, like monotonicity, appear to be of great relevance. We first focus on monotonicity, extendability and persistence of quantifiers. They are investigated in context of epistemological verifiability of scientific hypotheses. In the second part we show that some of these properties imply learnability. As a result two strong paradigms are joined: the paradigm of computational epistemology (see e.g. [6, 5]), which goes back to the notion of identification in the limit as formulated in [4], and the paradigm of investigating natural language determiners in terms of generalized quantifiers in finite models (see e.g.[10]).*

***Keywords:*** *identification in the limit, induction, monadic quantifiers, monotonicity, semantics learning, verification.*

## 1 Introduction

The 'identification in the limit' model [4] has found numerous applications in language learning analysis — for the most part in the acquisition of syntax. In contrast the model has been unappreciated in the investigations concerning learning of semantics.

On the other hand, in philosophy of science Gold's paradigm has been used to account for inductive reasoning and the process of approaching the correct theory about the world. In this domain various semantic properties of hypotheses are of great importance [6, 1].

*The author is the receiver of a Foundation for Polish Science Award for Young Researchers (START Programme 2008).

In the present paper we abstract from the distinction between learning and scientific inquiry. We hope that with this generality our results are relevant for both subjects. Our aim is to analyze semantic properties of inductive verifiability [6] and consider its connection with identification. The first section is devoted to two kinds of verifiability. The introduction of those notions is illustrated with the example of verifiability of monadic quantifiers in section 2. Next we present the basics about identification in the limit. In the culminating chapter 3 we compare the two notions. We conclude with theorems showing that with some restrictions certain types of verification imply identification.

## 2 Verification

The idea of verification, except for its obvious connections with semantics, is also very important in philosophy of science, where verifying and falsifying seem to be fundamental procedures for establishing an adequate theory and making predictions about the actual world. The semantic procedure of verification consists essentially in what follows:

**Verification task** Given model $M$ and a sentence $\varphi$, answer the question whether $M \models \varphi$.

Let us start with analyzing restrictions we should make on the verification task to be able to proceed with our considerations.

First of all, for the sake of generality we consider $M$ to be infinite. This allows us to talk about infinite procedures being successful in the limit. It is also very important to restrict our attention to computably enumerable structures. The reason is that we are interested in elements of the model being presented one by one — such an inductive procedure essentially requires that it is possible to enumerate them. In connection with this we also require that a presentation of a given model does not include repetitions. This restriction is made to simplify the procedure of counting elements without introducing any additional markers. We also have to say

something about $\varphi$ — the sentence involved in the verification task. We assume that $\varphi$ has the form of a quantifier sentence, with a quantifier closed under isomorphism. In other words, we assume that hypotheses of our framework are purely about cardinalities or relations between cardinalities, and not about the 'nature' of individual objects.

With the above-explained restrictions in mind, let us now move to define a formal framework of inductive verifiability.

**Definition 2.1** Let us consider a model $M = (U, B)$, where $U$ is an infinite, computably enumerable set, and $B \subseteq U$ is some computable unary predicate. Let us assume that $\lambda$ is an enumeration of the elements of $U$, without repetitions.

By 'environment of $M$', $\varepsilon$, we mean an infinite binary sequence such that: if $\lambda_n = x$, then $\varepsilon_n = \chi_B(x)$, where $\chi_B$ is the characteristic function of $B$. ◁

We will use the following notation:

$\varepsilon|n$ is the finite initial segment of $\varepsilon$ through position $n-1$ (i.e.: a sequence $\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_{n-1}$);

$SEQ$ denotes a set of all finite initial segments of all environments;

$set(\varepsilon)$ is a set of elements that occur in $\varepsilon$;

$h$ will refer to a hypothesis;

$C$ is a correctness relation between hypotheses and streams of data. $C(\varepsilon, h)$ is satisfied iff $h$ is correct with respect to $\varepsilon$, i.e., $h$ is true in the model represented by $\varepsilon$;

$\alpha$ is an assessment method — total map from hypotheses and finite data sequences to conjectures, $\alpha : H \times SEQ \to \{0, 1, !\}$.

Conjectures are outputs of $\alpha$; their meaning is the following:

- 1 — corresponds to the judgement that the hypothesis is true on the initial "up to now" segment of data;
- 0 — means that the hypothesis is judged to be false on the initial "up to now" segment of data;
- ! — appears as an announcement that there will be now mind change about the statement following in the next step (we also refer to it as the *eureka* sign).

## 2.1 Verification with Certainty

The first type of verification we want to discuss is verification with certainty. It holds when the process of verification is finished after a finite number of steps. We mean 'finished' in the sense that there is a point in the procedure at which the assessment method, $\alpha$, *decides* that the hypothesis, $h$, is true and that it can stop computing right there, because $h$ being false is no longer an option. In such a case we can informally say that $\alpha$ is 'sure' or 'certain' about the answer. This is where the name 'verification with certainty' comes from.

Formally, we will require that the step when certainty comes into the picture is marked with the *eureka* symbol '!' and the actual content of this certainty — the hypothesis being true or false — is '1' or '0', respectively, answered in the next step.

Let us first introduce the general notion of producing an answer with certainty.

**Definition 2.2** We say that $\alpha$ produces $b$ with certainty on $(h, \varepsilon)$ iff there is an $n$ such that:

1. $\alpha(h, \varepsilon|n) =!$, and
2. $\alpha(h, \varepsilon|n+1) = b$,
3. for each $m < n$, $\alpha(h, \varepsilon|m) \neq !$, and
4. all values after $n+1$ are irrelevant.

◁

Verification and falsification with certainty are defined as an adequate production of 0 or 1 with certainty, respectively.

**Definition 2.3** We say that $\alpha$ verifies $h$ with certainty on $\varepsilon$ (with respect to $C$) iff $\alpha$ produces 1 with certainty on $(h, \varepsilon) \Leftrightarrow C(\varepsilon, h)$. Definition of refutation with certainty is analogous. ◁

**Definition 2.4** We say that $h$ is verifiable with certainty iff there is an $\alpha$, which for each $\varepsilon$ verifies $h$ on $\varepsilon$ with certainty iff $h$ is true on $\varepsilon$. ◁

Verification with certainty satisfies the condition of positive introspection of knowledge, i.e., as soon as $\alpha$ answers '!' on $h$, it 'knows' the logical value of $h$. Such a situation does not occur in verification in the limit, which is defined below.

## 2.2 Verification in the Limit

Verification in the limit is much weaker than verification with certainty. In order to define it we exclude the *eureka* sign '!' from the set of possible answers. We restrict the power of the verification procedure $\alpha$ in such a way that it can give only two answers:

1 — corresponds to the fact that the hypothesis is judged to be true on the initial "up to now" segment of data;

0 — the hypothesis is judged to be false on the initial "up to now" segment of data.

As in the previous case, this type of verification consists in giving partial answers to finite initial segments of the environment. This time however the procedure is endless. We are dealing here with an infinite sequence of answers. We say that a procedure verifies a hypothesis in the limit if and only if there is a step in the procedure where the answer is 1 and it stays that way for the rest of the computation.

**Definition 2.5** We say that $\alpha$ verifies a hypothesis, $h$, in the limit iff:

$$\exists n \forall m > n \; \alpha(h, \varepsilon|m) = 1.$$

◁

**Definition 2.6** We say that $h$ is verifiable in the limit iff there is an $\alpha$, which for each $\varepsilon$ verifies $h$ in the limit on $\varepsilon$ iff $h$ is true on $\varepsilon$. ◁

In the general case of verification in the limit the fact of verification is not 'visible' to $\alpha$. Whether a hypothesis has been verified can be judged only from a global perspective. Limiting verification corresponds to the scientific strategy of claiming adequacy of some 'up to now' correct hypothesis as long as possible. There is no guarantee however that in the light of future data it will not be rejected. When dealing with verifiability in the limit a scientist has to remain alert all the time.

# 3 Application: Verification of Monotone Quantifiers

The restriction made in the previous section, that hypotheses of our framework are purely about cardinalities or relations between cardinalities, and not about the 'nature' of individual objects leads us to treat hypotheses as generalized quantifiers. Informally speaking a given hypothesis can be identified with the class of models in which it is true. The same works for quantifiers. Even if intuitively quantifiers are formal counterparts of (natural language) determiners, we have a theory of generalized quantifiers which instructs us to reduce a quantifier simply to the class of models in which this quantifier is true. So, running the risk of being charged with philosophical insensitivity, we will use the notions of quantifiers and hypotheses interchangeably.

In order to talk about the properties we are interested in we have to provide the relational definition of generalized quantifier.

**Definition 3.1** A generalized quantifier $\mathsf{Q}$ of type $t = (n_1, \ldots, n_k)$ is a functor assigning to every set $M$ a $k$-ary relation $\mathsf{Q_M}$ between relations on $M$ such that if $(R_1, \ldots, R_k) \in \mathsf{Q_M}$ then $R_i$ is an $n_i$-ary relation on $M$, for $i = 1, \ldots, k$.

◁

It is quite prevalent in the philosophical literature to link notions of verifiability (with certainty) and falsifiability (with certainty) to the existential and universal quantifier, respectively. In fact, as we are going to see, this intuitive correspondence includes a broader class with quantifiers of some special monotonicity properties. We will discuss this connection below.

## 3.1 Quantifiers of Type $\langle 1 \rangle$

Let us now focus on properties of generalized quantifiers of type $\langle 1 \rangle$. First we define what it means for a quantifier to be monotone increasing and extendable.

**Definition 3.2**

**(MON↑)** We say that a quantifier $\mathsf{Q_M}$ of type $\langle 1 \rangle$ is monotone increasing (MON↑) iff the following holds: if $A \subseteq A' \subseteq M$, then $\mathsf{Q_M}(A)$ implies $\mathsf{Q_M}(A')$.

**(EXT)** A quantifier $\mathsf{Q}$ of type $\langle 1 \rangle$ satisfies EXT iff for all models $M$ and $M'$: $A \subseteq M \subseteq M'$ implies $\mathsf{Q_M}(A) \implies \mathsf{Q_{M'}}(A)$.

◁

In other words, monotonicity guarantees that extending the predicate does not change the logical value of the quantifier from true to false. On the other hand extension ensures that adding new elements to the complement of $A$ does not make a true quantifier false.

Comparison of the notions of verifiability with certainty and monotonicity allows us to state the following proposition:

**Proposition 3.3** *Let* $\mathsf{Q}$ *be a* MON↑ *and* EXT *quantifier of type* $\langle 1 \rangle$. *There exists a model* $M = (U, A)$ *with finite* $A \subseteq U$ *such that* $\mathsf{Q}_M(A)$ *iff* $\mathsf{Q}$ *is verifiable with certainty for arbitrary computably enumerable models*.

**Proof.** ($\Rightarrow$) Let us first assume that $\mathsf{Q}$ of type $\langle 1\rangle$ is MON$\uparrow$ and EXT, and that there exists a model $M = (U, A)$ with finite $A \subseteq U$ such that $\mathsf{Q}_M(A)$. We use the characteristic function of $A$, $\chi_A$, to get an infinite sequence, $\varepsilon_A$, of 0's and 1's representing $M$. $\varepsilon_A$ is an environment of $M$. We run the $\alpha$ procedure on $\varepsilon_A$ and $\mathsf{Q}(A)$. Step by step, while being fed, $\alpha$ constructs a model $M' = (U', A')$. This happens in the following way.

First we take $n := 0, U' := \emptyset, A' := \emptyset$.

$\alpha$ reads $\varepsilon_n$: if $\varepsilon_n = 1$, then $|A'| := |A'| + 1$; else $|\bar{A}'| := |\bar{A}'| + 1$. $\alpha$ checks if $\mathsf{Q}(A')$: if it holds, $\alpha$ answers '!' and 1 to the rest of $\varepsilon_A$; otherwise it answers 0 and moves to $n := n + 1$.

The procedure $\alpha$ verifies $\mathsf{Q}(A)$ with certainty. This is because $\mathsf{Q}(A)$ is true in $M$, and from assumptions about MON $\uparrow$ and EXT, we know there is a finite cardinality of $A'$ which satisfies $\mathsf{Q}(A')$. As soon as $\alpha$ reaches this cardinality there is no possibility that $\mathsf{Q}(A)$ changes its logical value at an extension $A'$, $\bar{A}'$ in $M'$.

($\Leftarrow$) Let us assume that $M \models \mathsf{Q}(A)$, and that there is a procedure $\alpha$ which verifies with certainty on $\varepsilon_A$. Therefore, there is a point, $n$, at which $\alpha$ answers ! and then 1. Then we know that $\mathsf{Q}(A')$, where $|A'|$ is equal to the number of 1s in $\varepsilon_A|n$ and $|\bar{A}'|$ is equal to the number of 0s in $\varepsilon_A|n$. What remains of $\varepsilon$ is not relevant for the logical value of $\mathsf{Q}(A')$. This means that if $A' \subseteq A''$ then $\mathsf{Q}(A'')$ and if $M' \subseteq M''$ then $Q_{M''}(A')$. This is the same as saying that $\mathsf{Q}$ is MON$\uparrow$ and EXT. QED

Having this in mind we can also consider which type $\langle 1\rangle$ quantifiers correspond to the notion of falsifiability with certainty. The answer is as follows:

**Proposition 3.4** *Let $\mathsf{Q}$ be a quantifier of type $\langle 1\rangle$. $\mathsf{Q}$ is verifiable with certainty iff $\neg\mathsf{Q}$ is falsifiable with certainty.*

**Proof.** ($\Rightarrow$) First assume that $\mathsf{Q}$ is verifiable with certainty. That is: there is a procedure $\alpha$ such that for every model $M$ if $M \models \mathsf{Q}(A)$, then $\alpha$ verifies $\mathsf{Q}(A)$ with certainty. We now construct a procedure $\alpha'$ such that it falsifies $\neg\mathsf{Q}$ with certainty.

$$\alpha'(\varepsilon_A|n) = \begin{cases} 1 & \text{if } \alpha(\varepsilon_A|n) = 0, \\ 0 & \text{if } \alpha(\varepsilon_A|n) = 1, \\ ! & \text{if } \alpha(\varepsilon_A|n) = \ !. \end{cases}$$

Since $\neg\mathsf{Q}$ is a complement of $\mathsf{Q}$, this procedure falsifies $\neg\mathsf{Q}$ on $A$ iff $\neg\mathsf{Q}$ is false in $M$. ($\Leftarrow$) The other direction works the same way. QED

## 3.2 Quantifiers of Type $\langle 1, 1\rangle$

In the linguistic context it is common to investigate quantifiers of type $\langle 1, 1\rangle$. It is often assumed (see e.g. [9]) that all natural language determiners correspond to so-called CE-quantifiers. CE-quantifiers satisfy three requirements: isomorphism closure (ISOM), extension and conservativity (CONS). (EXT) for quantifiers of type $\langle 1, 1\rangle$ is a natural extension of the definition for type $\langle 1\rangle$. Below we define (CONS).

**Definition 3.5** We call a quantifier $\mathsf{Q}$ of type $\langle 1, 1\rangle$ conservative iff:

**(CONS)** $\forall A, B \subseteq M$: $\mathsf{Q}_M(A, B) \iff Q_M(A, A \cap B)$.

◁

CE-quantifiers then have the property that their logical value depends only on the cardinality of the two constituents, $A - B$ and $A \cap B$, in the model. The part of $B$ falling outside of the scope of $A$ does not influence the logical value of a CE-quantifier. For the rest of the present section we will restrict ourselves to CE-quantifiers.

We will also need a notion of left-side monotonicity, which is usually called 'persistence'.

**Definition 3.6** We call a quantifier $\mathsf{Q}$ of type $\langle 1, 1\rangle$ persistent iff:

**(PER)** If $A \subseteq A' \subseteq M$ and $B \subseteq M$, then $\mathsf{Q_M}(A, B) \Rightarrow \mathsf{Q_M}(A', B)$.

◁

Persistence guarantees that adding new elements to both important constituents $A$ and $A \cap B$ does not change the logical value of the quantifier from true to false.

We claim the following:

**Proposition 3.7** *Let $\mathsf{Q}$ be a* PER CE*-quantifier of type $\langle 1, 1\rangle$. There exists a model $M = (U, A, B)$ such that $A \cap B$ is finite and $\mathsf{Q}_M(A, B)$ iff it is verifiable with certainty.*

**Proof.** The proof is analogous to the proof of Proposition 1. We simply focus on two constituents of the model: $A - B$ and $A \cap B$, and treat them as $\bar{A}$ and $A$ (respectively) in the proof of Proposition 1. QED

**Proposition 3.8** *Let $\mathsf{Q}$ be a* CE*-quantifier of type $\langle 1, 1\rangle$. $\neg\mathsf{Q}$ is falsifiable with certainty iff $\mathsf{Q}$ is verifiable with certainty.*

**Proof.** Analogous to the the proof of Proposition 2. QED

Monotonicity has so far given some explanation for differences in the comprehension of quantifiers. The distinction between verifiability and refutability of quantifiers provides new thoughts regarding this problem. It gives some additional psychologically plausible explanation for differences in the difficulty of natural language quantifiers. It can

be argued that verification of hypotheses is much easier for people than refutation. In consequence verifiable quantifiers can be considered much easier in natural language processing than refutable ones.

## 4 Identifiability through Verification

Historically speaking, philosophical analysis of the scientific discovery process led to skepticism. It has been claimed that its creative content cannot be accounted for by any scientific means, in particular by no mathematical or algorithmic model [2]. The natural situation of discovery is indeed so complex and non-uniform that it seems impossible to catch it in an adequate formalism. However, some approximations, which to a certain extent idealize the process, are not only makable, but also already existing and are ready to use. The framework of identification in the limit proposed in [4] started a long line of mathematical investigation of the process of language learning. At first sight scientific discovery and learning might seem distant from each other. In the present paper we assume the adequacy of the identification model for scientific inquiry analysis (for similar approaches see: [6, 7]).

Intuitively, the verification procedure (discussed in the previous section) is a part of scientific discovery. The latter can be seen as a compilation of assuming hypotheses, checking their logical value on data, and changing them to another hypothesis, if needed. In the present section we will introduce the identification formalism and present some ideas and facts about its correspondence to verification.

### 4.1 Identification

The identification in the limit approach [4] gives a mathematical reconstruction of the process of inductive inference. The task consists in guessing a correct hypothesis on the basis of an inductively given, infinite sequence of data about the world.

The framework includes: a class of hypotheses $H$, an infinite sequence of data about the world $\varepsilon$, a learning function $f$ (a scientist).

We will explain the general idea of identification in the limit in terms of a simple game between a Scientist and Nature. First, some class of hypotheses, $H$, is chosen. It is known by both players. Then Nature chooses a single hypothesis, $h$, from $H$, to correctly describe the actual world. Then Nature starts giving out atomic information about the world. She does this in an inductive way. Each time the Scientist gets a piece of information, he guesses a hypothesis from the previously defined class on the basis of the sequence of data given so far. Identification in the limit is successful, if the guesses of the Scientist after some finite time stabilize on the correct answer.

Let us now specify the elements of the framework. By hypotheses we again mean quantified formulae, with a logical (closed under isomorphism) quantifier of type $\langle 1 \rangle$ or CE-quantifier of type $\langle 1, 1 \rangle$ (see e.g. [9]). The reason for this is the same as in the case of verification — that we want order- and intension-independent hypotheses, and a clear and relevant binary representation of models. The above-mentioned encoding of models serves as a basis for environments. The learning function, also referred to as the 'scientist', is defined as $f : SEQ \to H$.

**Definition 4.1** [Identification in the limit]
We say that a learning function, $f$:

1. identifies $h \in H$ on $\varepsilon$ for $M \models h$ in the limit iff for cofinitely many $n$, $f(\varepsilon|n) = h$.

2. identifies $h \in H$ in the limit iff it identifies $h$ in the limit on every $\varepsilon$ for every $M$, such that $M \models h$.

3. identifies $H$ in the limit iff it identifies in the limit every $h \in H$.

$\triangleleft$

We can analogously define the much stronger notion of identifiability with certainty. The difference is that in this case the learning function 'knows' when it has identified the correct hypothesis.

**Definition 4.2** [Identification with certainty]
We say that a learning function, $f$:

1. identifies $h \in H$ with certainty on $\varepsilon$ for $M \models h$ iff for some $n$, $f(\varepsilon|n) =!$ and $f(\varepsilon|n+1) = h$.

2. identifies $h \in H$ with certainty iff it identifies $h$ with certainty on every $\varepsilon$ for every $M \models h$.

3. identifies $H$ with certainty iff it identifies with certainty every $h \in H$.

$\triangleleft$

### 4.2 Comparing verification and identification

In the present section we will state two theorems. They show a connection between identifiability and verifiability.

#### 4.2.1 Certainty setting

Let us take a class of hypotheses, $H$, and the sequence, $\varepsilon$, of data about the actual world. Assume that $H$ contains only mutually disjoint hypotheses verifiable with certainty, i.e., for every $h \in H$ there is a procedure $\alpha$, which verifies $h$ with certainty iff it is true in the actual world.

**Theorem 4.3** *Every such computably enumerable class $H$ is identifiable with certainty.*

**Proof.** Assume that $H$ is a computably enumerable class of mutually disjoint hypotheses verifiable with certainty. We define a procedure **Id-Cert** which identifies with certainty every hypothesis from the class $H$. An example of a run of the procedure is presented in Figure 1.

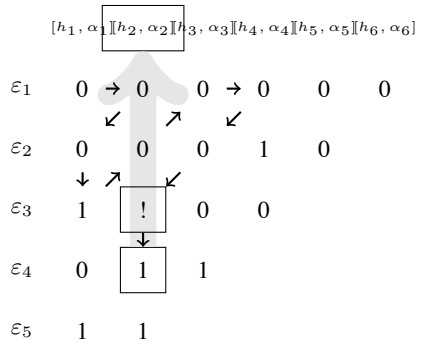


**Figure 1. Identifiability with certainty**

Since $H$ is computably enumerable we can assume existence of a sequence $(h)_n$ which enumerates $H$. Each $h_n$ is associated with its verification with certainty procedure $\alpha_n$. **Id-Cert** works in the following way: it first checks $\alpha_1(h_1, \varepsilon_1)$ (the value of the first hypothesis on the first piece of data), then it proceeds according to the diagonal enumeration of $\alpha_n(h_n, \varepsilon_m)$ until it meets '!'. Then it performs a check for $\alpha_n(h_n, \varepsilon_{m+1})$. If $\alpha_n(h_n, \varepsilon_{m+1}) = 1$, then **Id-Cert** stops and answers $h_n$. Otherwise it moves back to $\alpha_n(h_n, \varepsilon_m)$ and continues to perform the diagonal procedure.

By assumption every $h \in H$ is verifiable with certainty. Therefore if $h_n$, for some $n$, is true on $\varepsilon$, then $\alpha_n$ will eventually produce '!'. And since **Id-Cert** performs a diagonal search it does not miss any answer. Hence, **Id-Cert** identifies every $h \in H$ with certainty, so $H$ is identifiable with certainty. QED

Let us again take a class of hypotheses, $H$, and the sequence, $\varepsilon$, of data about the actual world. Assume that $H$ contains only hypotheses verifiable with certainty, but this time let us drop the assumption of $H$ being a class of mutually disjoint hypotheses. Then we can prove what follows.

**Theorem 4.4** *Every such computably enumerable class $H$ is identifiable in the limit.*

**Proof.** The proof is very similar to the proof of the previous theorem. We use the same diagonal method. This time however identification does not stop on the first '!' it encounters. Let us assume that '!' happens for $\varepsilon_n$. Instead, it answers the relevant $h$: the hypothesis which was first recognized to be verified with certainty; then it goes on with the diagonal search looking for a hypothesis, $h'$, which reveals '!' for some $\varepsilon_m$, where $m < n$. If it meets such an $h'$ it keeps answering it as long as no other 'better fitting' hypothesis is found. An example of a run of the procedure is presented in Figure 2.

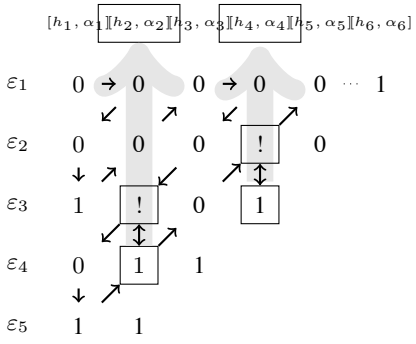


**Figure 2. Identifiability with certainty**

By assumption every $h \in H$ is verifiable with certainty. Therefore if $h_n$, for some $n$, is true on $\varepsilon$, then $\alpha_n$ will eventually produce '!'. And since this identification performs a diagonal search it does not miss any answer. Hence every $h \in H$ is identified in the limit, so $H$ is identifiable in the limit. QED

#### 4.2.2 Limiting setting

Let us again take a computably enumerable class of mutually disjoint hypotheses, $H$, and a sequence, $\varepsilon$, of data about the actual world. But this time let us agree that $H$ consists of hypotheses that are verifiable in the limit, i.e., for every $h \in H$ there is a procedure $\alpha$ which verifies $h$ in the limit iff $h$ it is true.

**Theorem 4.5** *Every such computably enumerable class $H$ is identifiable in the limit.*

**Proof.** Assume that $H$ is a computably enumerable class of mutually disjoint hypotheses that are verifiable in the limit. This means that for every $h_n \in H$ there is a procedure $\alpha_n$ which verifies $h$ in the limit if and only if $h$ is true. We are now going to define a procedure **Id-Lim** which identifies

every hypothesis from the class $H$. An example of a run of the **Id-Lim** is presented in Figure 3.

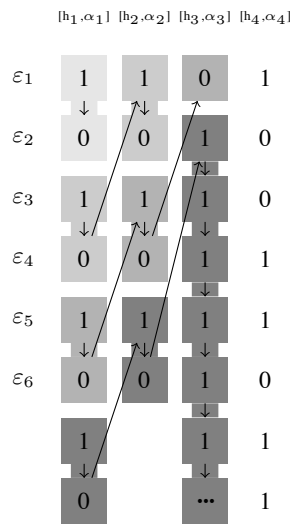


**Figure 3. Id-Lim identifiability**

Since $H$ is computably enumerable we can assume the existence of the sequence $(h)_n$ enumerating the hypotheses from $H$. Each of them is associated with its verification in the limit procedure $\alpha_n$.

The algorithm **Id-Lim** first performs a single check for $\{h_1\}$:

If $\alpha_1(h_1, \varepsilon|1) = 1$, then **Id-Lim** outputs $h_1$ and moves to $\alpha_1(h_1, \varepsilon|2)$. The answer is repeated until there is an $n$ such that $\alpha_1(h_1, \varepsilon|n) = 0$. In this case it starts the test for $\{h_1, h_2\}$, i.e., starting from $\varepsilon|n+1$ it looks for another 0 in the column $(h_1, \alpha_1)$ answering $h_1$ as long as $\alpha_1$ answers 1. When 0 is visited **Id-Lim** moves to $\alpha_2(h_2, \varepsilon_1)$ and performs a single check for $h_2$. In such manner we try to check $\{h_1\}$, $\{h_1, h_2\}$, $\{h_1, h_2, h_3\}$, ...

Procedure **Id-Lim** never stops. It is successful if after some point its guesses are still the same and correct with respect to $\varepsilon$.

Why does **Id-Lim** work? One can easily observe that **Id-Lim** runs through every finite sequence of 1s. Visiting a point in which $\alpha_n(h_n, \varepsilon_m) = 1$, it answers $h_n$. If there is a true hypothesis in $H$, **Id-Lim** will eventually enter an infinite sequence of 1s (in column $(h_m, \alpha_m)$, say), since $H$ consists of hypotheses verifiable in the limit. Once it enters this sequence there is no way out — **Id-Lim** will indefinitely answer $h_m$. Therefore **Id-Lim** identifies every $h \in H$ in the limit, and hence $H$ is identifiable in the limit. QED

In case **Id-Lim** identifies some $h_n$ the procedure needs to remember a finite but not predetermined number of points in $\varepsilon$. We would like to have an algorithm which does not run back and forth on the environment. The answer to this is procedure which is introduced below. Let us call it **Id-Lim**$^*$. For this procedure it is enough to remember only one point, namely the position in which the procedure finds itself at each moment.

**Id-Lim**$^*$ uses essentially the same idea of column-ruled searching for strings of 1s. It also consecutively performs it for $\{h_1\}$, $\{h_1, h_2\}$, $\{h_1, h_2, h_3\}$, ...The difference is that when it eventually leaves one column, starting a test for a new hypothesis, it does not go back to $\varepsilon_1$. Instead, it simply moves to the value in the next column but in the same row.

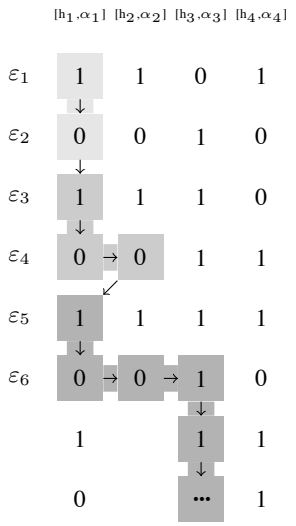


**Figure 4. Id-Lim$^*$ identifiability**

The difference between **Id-Lim** and **Id-Lim**$^*$ is mainly in the use of $\varepsilon$. With **Id-Lim**$^*$ it is enough to run through $\varepsilon$ once without going back. In case of **Id-Lim** every time we fail on some hypothesis and enter a new one, previously not visited, it has to start reading $\varepsilon$ from the beginning. **Id-Lim**$^*$ also identifies $H$. It simply leaves out the truth values of hypotheses on some already visited initial segment of $\varepsilon$.

## 5 Conclusion

The approach presented in this paper can be seen as an attempt to find some general semantic correlates of identification. Inductive verification can be treated as a condition for and a part of the identification process. This fact contributes to the general problem of semantics learning and to modeling the process of scientific inquiry.

Some attempts to approach the problem of learning of semantic constructions are already present in the literature [8, 3]. What is the connection with this framework? The present approach has much to do with the more general idea

of model-theoretic learning [1, 7], but it is also related to the work of H.-J. Tiede [8]. In his, slightly different, framework he shows that the class of first-order definable persistent quantifiers of type $\langle 1, 1\rangle$ is identifiable in the limit. This result is consistent with our considerations. In fact, for the same class of quantifiers we show that it is verifiable with certainty, and that each class containing solely verifiable with certainty structures is identifiable in the limit.

Intuitively there are at least two main parts of human semantic competence. One of them is responsible for producing grammatically correct (syntax domain) or true (semantics domain) hypotheses. The second is a natural correlate of model-checking, i.e., the competence of deciding whether a sentence is true or false in the actual world. The results presented in this paper show how the latter can be embedded in the identification (learning or discovering) process. In this light verification can be seen as a pillar of learning abilities.

## References


[1] J. van Benthem and A. Ter Meulen, editors. *Handbook of Logic and Language*. MIT Press, Cambridge, MA, USA, 1997.

[2] P. Feyerabend. *Against Method*. Verso Press, London, 1975.

[3] N. Gierasimczuk. The problem of learning the semantics of quantifiers. In B. ten Cate and H. Zeevat, editors, *Logic, Language, and Computation, TbiLLC 2005*, volume 4363 of *Lecture Notes in Artificial Intelligence*, pages 117–126. Springer, 2007.

[4] E. Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.

[5] S. Jain, D. Osherson, J. S. Royer, and A. Sharma. *Systems that Learn*. MIT Press, Chicago, 1999.

[6] K. Kelly. *The Logic of Reliable Inquiry*. Oxford University Press, Oxford, 1996.

[7] E. Martin and D. Osherson. *Elements of Scientific Inquiry*. MIT Press, Cambridge, 1998.

[8] H.-J. Tiede. Identifiability in the limit of context-free generalized quantifiers. *Journal of Language and Computation*, 1:93–102, 1999.

[9] J. Väänänen. On the expressive power of monotone natural language quantifiers over finite models. *Journal of Philosophical Logic*, 31:327–358, 2002.

[10] J. van Benthem. *Essays in Logical Semantics*. D. Reidel, Dordrecht, 1986.