## 0.1 Identification in the limit from positive data

The box below precisely defines the paradigm when learning from *positive* data. Let us define the "evidence" when learning from positive data more precisely. A **positive presentation** of a stringset $S$ is a function $\varphi : \mathbb{N} \to S$ such that $\varphi$ is onto. Recall that a function $f$ is onto provided for every element $y$ in its co-domain there is some element $x$ in its domain such that $f(x) = y$. Here, this means for every string $s \in S$, there is some $n \in \mathbb{N}$ such that $\varphi(n) = s$.

**Definition 1** (Identification in the limit from positive data)**.**

```
1 Algorithm $A$ identifies in the limit from positive data a class of stringsets $C$ provided
2     for all stringsets $S \in C$,
3         for all positive presentations $\varphi$ of $S$,
4             there is some number $n \in \mathbb{N}$ such that
5                 for all $m > n$,
6                     • the program output by $A$ on $\varphi\langle m\rangle$ is the same as the the program
7                       output by $A$ on $\varphi\langle n\rangle$, and
8                     • the program output by $A$ on $\varphi\langle m\rangle$ solves the membership problem
9                       for $S$.
```

Here is breakdown of what these lines mean.

**Line 1** Establishes the name of the relationship between an algorithm $A$ and a collection of stringsets $C$ provided the definition holds.

**Line 2** The algorithm must succeed for all $S \in C$.

**Line 3** The algorithm must succeed for all positive presentations $\varphi$ of $S$.

**Line 4** It succeeds on $\varphi$ for $S$ if there is a point in time $n$

**Line 5** such that for all future points in time $m$,

**Lines 6-7** the output of $A$ converges to the same program, and

**Lines 8-9** the output of $A$ correctly solves the membership problem for $S$.

This paradigm is also called **learning from text.**

## 0.2 Classes of Languages

We have already seen that the following classes of languages are identifiable in the limit from positive data.

1. BAR-X $= \{\bar{x} \mid x \in \Sigma^*\}$. Recall $\bar{x} \overset{\text{def}}{=} \{w \in \Sigma^* \mid w \neq x\}$.

2. For each $k \in \mathbb{N}$, $\mathrm{SP}_k$

3. For each $k \in \mathbb{N}$, $\mathrm{SL}_k$

The following classes of stringsets are fundamental ones in formal language theory so it makes sense to be curious about their learnability.

1. The class of finite stringsets (FIN).

2. The class of regular stringsets (REG).

3. The class of context-free stringsets (CF).

4. The class of context-sensitive stringsets (CS).

These are in the following relationship: FIN $\subsetneq$ REG $\subsetneq$ CF $\subsetneq$ CF.

## 0.3 Results

**Theorem 1.** *FIN is identifiable in the limit from positive data.*

**Exercise 1.** Prove this theorem. (Hint: FIN can be learned with string extension learning.)

Any class which includes every finite language and at least one more is a *superfinite* class of languages.

**Theorem 2.** *No superfinite class of stringsets is identifiable in the limit from positive data.*

There are different ways to prove this theorem. Here is one based on (de la Higuera, 2010, p. 151).

**Proof**[sketch] Consider any superfinite class of languages $C$. By definition $C$ includes all finite languages and at least one infinite language $L_\infty$.

Let $x_1, x_2, \ldots$ be the infinitely many words of $L_\infty$.

Let $L_1 = \{x_1\}$, $L_2 = L_1 \cup \{x_2\}$, $L_3 = L_2 \cup \{x_3\}$, and so on. So $L_k = L_{k-1} \cup \{x_k\}$. For each $k \in \mathbb{N}$, $L_k \in C$ since $L_k$ is finite.

For the sake of contradiction, assume there is an algorithm $A$ that identifies $C$ in the limit from positive data. We will show there is a presentation for $L_\infty$ for which $A$ fails to converge.

Pick a presentation $\varphi_1$ for $L_1$. Since $A$ identifies $L_1$ in the limit, there is a convergence point $i_1$ such that $A$ outputs a grammar for $L$ on $\varphi_1[i_1]$. Let $\varphi_2$ be some presentation of $L_2$ such that for all $j < i_1$, $\varphi_2(j) = \varphi_1(j)$ and $\varphi_2(i_1 + 1) = x_2$. More generally, let $\varphi_k$ be some presentation of $L_k$ such that for all $j < i_{k-1}$, $\varphi_k(j) = \varphi_{k-1}(j)$ and $\varphi_k(i_{k-1} + 1) = x_k$.

In this manner we construct a presentation $\varphi_\infty$ for $L_\infty$. Consider any $i \in \mathbb{N}$. There exists $j, j+1$ such that $i_j < i \leq i_{j+1}$. Let $k$ equal $j + 1$. Then $\varphi_\infty(i)$ equals $\varphi_k(i)$.

How does $A$ behave on $\varphi_\infty$? It does not converge. This is because for all $k \in \mathbb{N}$, at time point $i_k$, $A$ will output a program for $L_k$. So it never converges to a grammar for $L_\infty$ even though $\varphi_\infty$ is a positive presentation for $L_\infty$. □

Gold explains the idea behind his result this way.

> It is of great interest to find why information presentation by text is so weak and under what circumstances it becomes stronger. Therefore, it is worthwhile to understand the method used in Theorems I.8 and I.9 to prove that any class of languages containing all finite languages and at least one infinite language is not identifiable in the limit from a text in five out of six of the models using text.
>
> The basic idea is proof by contradiction. Consider any proposed guessing algorithm. It must identify any finite language correctly after a finite amount of text. This makes it possible to construct a text for the infinite language which will fool the learner into making a wrong guess an infinite number of times as follows. The text ranges over successively larger, finite subsets of the infinite language. At each stage it repeats the elements of the current subset long enough to fool the learner. Thus, the method of proof of the negative results concerning text depends on the possibility of there being a huge amount of repetition in the text. Perhaps this can be prevented by some reasonable probabilistic assumption concerning the generation of the text. In this case one would only require identification in the limit with probability one, rather than for every allowed text.
>
> I have been asked, "If information presentation is by means of text, why not guess the unknown language to be the simplest one which accepts the text available?" This is identification by enumeration. It is instructive to see why it will not work for most interesting classes of languages: The universal language (if it is in the class) will have some finite complexity. If the unknown language is more complex, then the guessing procedure being considered will always guess wrong, since the universal language is consistent with any finite text. This follows from the fact that, if L is the unknown language and if L' ⊃ L, then L' is consistent with any finite segment of any text for L. The problem with text is that, if you guess too large a language, the text will never tell you that you are wrong.

It immediately follows that the class of regular, context-free, context-sensitive, and computably enumerable classes of stringsets are not identifiable in the limit from positive data.

Furthemore, for every finite stringset $S$, there is some $k$ such that $S$ is Strictly $k$-Local. Thus FIN $\subsetneq$ SL. Hence neither SL nor LT nor LTT nor TSL is identifiable in the limit from positive data.

**Theorem 3** ((Angluin, 1980)). *A class $C$ is identifiable in the limit from positive data iff for each $S \in C$ there is a finite set $D \subseteq S$ such that for all $S' \in C$ such that $D \subseteq S'$ it holds that $S' \not\subsetneq S$.*

Pictorially, Figure 1 is the situation that cannot obtain.

**Corollary 1.** *Every finite class of languages is identifiable in the limit from positve data.*

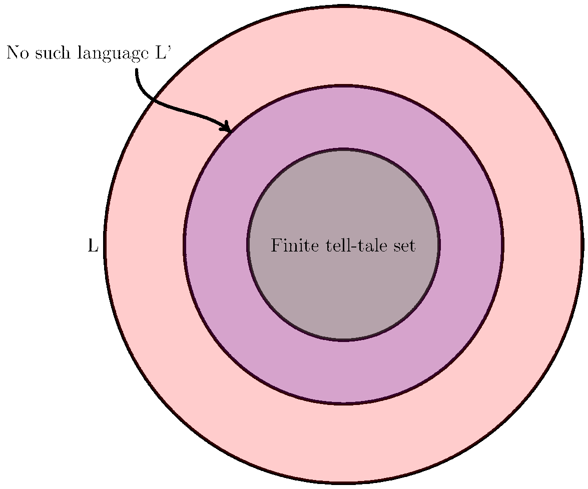

Figure 1: No such L' in every class identifable in the limit from positive data!

Gold's theorems and Angluin's theorems above are the basis for the so-called "Subset problem" in linguistics literature on learning (Wexler and Culicover, 1980; Berwick, 1985).

# 1 Identification in the limit from positive and negative data

A **positive and negative presentation** of a stringset $S$ provides example strings not in $S$ in addition to example strings in $S$. This can be formalized using the *characteristic function* of $S$. Every set $S$ has a characteristic function with domain $\Sigma^*$ defined as follows.

$$f_S(s) = \begin{cases} 1 & \text{iff } s \in S \\ 0 & \text{otherwise} \end{cases}$$

Characteristic functions are total functions, which means defined for all $s \in \Sigma^*$. Also recall, that we write $(x, y) \in f$ whenever $f(x) = y$. So we can think of $f_S$ as a set of points where $(s, 0)$ means $s \notin S$ and $(s, 1)$ means $s \in S$.

Then a **positive and negative presentation** of a stringset $S$ is a function $\varphi : \mathbb{N} \to f_S$ such that $\varphi$ is onto. Here, this means for every string $s \in \Sigma^*$, there is some $n \in \mathbb{N}$ such that $\varphi(n) = (s, f_s(s))$.

**Definition 2** (Identification in the limit from positive and negative data).

```
1  Algorithm $A$ *identifies in the limit from positive and negative data* a class of stringsets
2  $C$ provided
3        for all stringsets $S \in C$,
4              for all positive and negative presentations $\varphi$ of $S$,
5                    there is some number $n \in \mathbb{N}$ such that
6                          for all $m > n$,
7                          • the program output by $A$ on $\varphi\langle m\rangle$ is the same as the the program
8                            output by $A$ on $\varphi\langle n\rangle$, and
9                          • the program output by $A$ on $\varphi\langle m\rangle$ solves the membership problem
10                           for $S$.
```

The only difference between the definition above and the one in Definition 1 is in line 3. This paradigm is also called **learning from an informant.**

**Theorem 4.** *The computable class of languages is identifiable in the limit from positive and negative data.*

**Proof**[sketch] The algorithm proceeds by enumeration over programs. Since programs are strings, we can essentially use the enumeration for strings we used before.

The learning algorithm finds the first program in the enumeration that successfully classifies all of the data it has observed so far in $\varphi$.

How does it do this? Well, it looks at the first program in the enumeration and submits to it each data point $\varphi[i]$. If the first program fails to compute anything, or classifies any one of the data points incorrectly, the learning algorithm moves to the next program and checks again. This repeats. Evenutally, it must find a program which classifies all of the observed data points correctly. At this point, it outputs this program.

How do we know this algorithm converges to a correct program for $S$? Well, there is a program for $S$ in the enumeration. There may be more than one such program so let $P$ be the first program in the enumeration for $S$. Once the learning algorithm reaches $P$ it will output $P$ since $P$ will classify all data points from $\varphi$ correctly because $\varphi$ is a positive and negative presentation of $S$.

How do we know the learning algorithm will eventually output $P$ on any positive and negative data presentation $\varphi$ for $S$? Consider any program $P'$ prior to $P$ in the enumeration. Since $P$ is the first program in the enumeration for $S$, $P'$ is not a program for $S$. It follows that there is some $w \in S$, or $w \notin S$ that $P'$ misclassifies. It follows that there is some point in time $i$ such that $\varphi(i) = (w, x)$ with $x \in \{0, 1\}$. At this point the learning algorithm will conclude that $P'$ does not classify everything it has seen in $\varphi[i]$ correctly, and will move to the next program in the enumeration. Since $P'$ was arbitrary, it follows that the algorithm will eventually reach adn output program $P$. □

## 2 Identification in the limit from primitive recursive texts

Recall that algorithms have to succeed for *any* text or from *any* informant. As we discuss later, this has been one source of criticism of Gold's learning paradigm.

Let's consider texts for the moment. That is, let's consider positive presentations for some stringset $S$ which has at least two strings in it. How many positive presentations are there? It should be easy to see that there are infinitely many. If $u, v$ are distinct words in $S$ then a text could start either $u, v$ or $u, u, v$ or $u, u, u, v$ or $u, u, u, u, v$ and so on. In fact, there are *uncountably many* presentations for $S$. This can be shown by the same diagnolization argument that we used earlier to show that there are uncountably many subsets of $\Sigma^*$.

We can also ask how many of these presentations are computable? Provided there are at least two strings in $S$, the answer is only countably many.

This situation is exactly analgous to the number of real numbers between 0 and 1, inclusive, and the number of computable real numbers between 0 and 1, inclusive.

In other words, *most* of the presentations that the Gold paradigm is required to succees on are *uncomputable*. Some have argued this is not reasonable.

Regardless of whether it is or not, we may be interested in what changes if we change the definition of learning to only require success on computable texts.

A particularly strong form of computability is computability via *primitive recursion*. This is weaker than Turing-machine computable. For example, Turing machines are not guaranteed to halt on input, but primitive recursive programs are guaranteed to halt. See Rogers (1967) for details on primitive recursion.

**Definition 3** (Identification in the limit from primitive recursive texts)**.**

```
1  Algorithm $A$ identifies in the limit from positive data a class of stringsets $C$ provided
2      for all stringsets $S \in C$,
3          for all positive, computable presentations $\varphi$ of $S$,
4              there is some number $n \in \mathbb{N}$ such that
5                  for all $m > n$,
6                  • the program output by $A$ on $\varphi\langle m\rangle$ is the same as the the program
7                    output by $A$ on $\varphi\langle n\rangle$, and
8                  • the program output by $A$ on $\varphi\langle m\rangle$ solves the membership problem
9                    for $S$.
```

The only difference between the definition above and the ones in Definitions 1 and 2 is in line 3.

**Theorem 5.** *The recursive (computable) class of languages is identifiable in the limit from primitive recursive texts.*

I'm not able at present to explain this proof, so it is omitted. I think the basic ideas are that (1) primitive recursive texts are enumerable and (2) it is possible to translate a primitive recursive text into a grammar a language equal to the content of the text (set of strings in the text). So an algorithm can identify by enumeration the primitive recursive text and thus the language the text is from.

## 3 Gold's interpretation of these results

From (Heinz, 2016):

> Gold (1967:453-454) provides three ways to interpret his three main results:
>
> 1. The class of natural languages is much smaller than one would expect from our present models of syntax. That is, even if English is context-sensitive, it is not true that any context-sensitive language can occur naturally...In particular the results on [identification in the limit from positive data] imply the following: The class of possible natural languages, if it contains languages of infinite cardinality, cannot contain all languages of finite cardinality.
> 2. The child receives negative instances by being corrected in a way that we do not recognize...
> 3. There is an a priori restriction on the class of texts [presentations of data; i.e. infinite sequences of experience] which can occur...
>
> The first possibility follows directly from the fact that no superfinite class of languages is identifiable in the limit from positive data. The second and third possibilities follow from Gold's other results on *identification in the limit from positive and negative data* and on *identification in the limit from positive primitive recursive data* ...
>
> Each of these research directions can be fruitful, if honestly pursued. For the case of language acquisition, Gold's three suggestions can be investigated empirically. We ought to ask
>
> 1. What evidence exists that possible natural language patterns form subclasses of major regions of the Chomsky Hierarchy?
> 2. What evidence exists that children receive positive and negative evidence in some, perhaps implicit, form?
> 3. What evidence exists that each stream of experience each child is exposed to is guaranteed to be generated by a fixed, computable process (i.e. computable probability distribution or primitive recursion function)? More generally, what evidence exists that the data presentations are a priori limited?
>
> My contention is that we have plenty of evidence with respect to question (1), some evidence with respect to (2), and virtually no evidence with respect to (3).

Finally, Gold concludes his paper this way.

> Concerning inductive inference, philosophers often occupy themselves with the following type of question: Suppose we are given a body of information and a set of possible conclusions, from which we are to choose one. Some of the conclusions are eliminated by the information. The question is, of the conclusions which are consistent with the in- formation, which is "correct"?
>
> If some sort of probability distribution is imposed on the set of conclusions, then the problem is meaningful. But if no basis for choosing between the consistent conclusions is postulated a priori, then inductive inference can do no more than state the set of consistent conclusions.
>
> The difficulty with the inductive inference problem, when it is stated this way, is that it asks, "What is the correct guess at a specific time with a fixed amount of information?" There is no basis for choosing between possible guesses at a specific time. However, it is interesting to study a guessing strategy. Now one can investigate the limiting behavior of the guesses as successively larger bodies of information are considered. This report is an example of such a study. Namely, in interesting identification problems, a learner cannot help but make errors due to incomplete knowledge. But, using an "identification in the limit" guessing rule, a learner can guarantee that he will be wrong only a finite number of times.

# 4 Criticisms

1. Identification in the limit from positive data is too hard. The texts can be adversarial.

2. identification in the limit doesn't address time or resource complexity of learning.

The first point is articulated well by Clark and Lappin (2011).

The second point is about feasibility. Learning by enumeration is very, very far from efficient. So even if every finite class of languages is identifiable in the limit from positive data, large finite classes may not be efficiently learnable because learning by enumeration is awfully slow! Similarly, Even if the recursive class is identifiable in the limit from primitive recursive text, it is not efficiently learnable. So we need some way to identify feasibly learnable subclasses.

Much research since Gold has aimed to incorporate feasibility into learning. The Probably Approximately Correct learning model is one influential example (Valiant, 1984; Anthony and Biggs, 1992; Kearns and Vazirani, 1994).

Many researchers advocate a learning setting where the aim is not to learn categorical stringsets but to learn probabilty distributions over them ("stochastic stringsets.") We will talk about this next.

The most repeated refrain ever in cognitive scince, computational linguistics about the theory of learning languages is this: "Gold (1967) showed that context-free grammars are not learnable but Horning (1969) showed that probabilistic context-free grammars are." There is so much confusion about this, I wrote about it: (Heinz, 2016).

# References

Angluin, Dana. 1980. Inductive inference of formal languages from positive data. *Information Control* 45:117–135.

Anthony, M., and N. Biggs. 1992. *Computational Learning Theory*. Cambridge University Press.

Berwick, Robert. 1985. *The acquisition of syntactic knowledge*. Cambridge, MA: MIT Press.

Clark, Alexander, and Shalom Lappin. 2011. *Linguistic Nativism and the Poverty of the Stimulus*. Wiley-Blackwell.

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

Heinz, Jeffrey. 2016. Computational theories of learning and developmental psycholinguistics. In *The Oxford Handbook of Developmental Linguistics*, edited by Jeffrey Lidz, William Synder, and Joe Pater, chap. 27, 633–663. Oxford, UK: Oxford University Press.

de la Higuera, Colin. 2010. *Grammatical Inference: Learning Automata and Grammars*. Cambridge University Press.

Horning, J. J. 1969. A study of grammatical inference. Doctoral dissertation, Stanford University.

Kearns, Michael, and Umesh Vazirani. 1994. *An Introduction to Computational Learning Theory*. MIT Press.

Rogers, Hartley. 1967. *Theory of Recursive Functions and Effective Computability*. McGraw Hill Book Company.

Valiant, L.G. 1984. A theory of the learnable. *Communications of the ACM* 27:1134–1142.

Wexler, Kenneth, and Peter Culicover. 1980. *Formal Principles of Language Acquisition*. MIT Press.