
The Learning Power of Evolution

Vitaly Feldman

IBM Almaden Research Center
650 Harry rd.
San Jose, CA 95120
vitaly@post.harvard.edu

Leslie G. Valiant*

Harvard University
33 Oxford st.
Cambridge, MA 02138
valiant@seas.harvard.edu

It has been widely recognized that learning and evolution have the commonality of involving adaptive processes that once started do not need a programmer or designer. It is tempting to seek some mystical extra power in evolution, beyond that of learning, simply because of the apparently spectacular consequences of evolution that we see around us. However, such approaches have not succeeded to date.

In response to this situation one of the authors made the, apparently radical, suggestion that evolution is nothing other than a *constrained* form of computational learning. In [Val08] a notion of evolvability was defined in a similar spirit to the definition of learnability. The goal of the definition is to offer a rigorous basis for the analysis of evolution and for distinguishing between *efficient evolution* and evolution that is only realized in some exponentially far limit.

Before summarizing this framework we describe the following motivating concrete instance. Consider the 20,000 or so genes in the human genome. For each such gene the condition under which the protein corresponding to it is expressed, in terms of all the other proteins, is encoded in its regulatory region. In other words each of the 20,000 or so proteins is controlled by a function f of the other 20,000 or so proteins. The issue here is that if the function f is restricted to too small a class then it will not be expressive enough to perform the complex functions of biology. On the other hand, if the function is an arbitrary function, or from a too extensive a class, then no evolutionary algorithm will exist to maintain the viability of this genetic network of functions as environmental conditions change. The goal of this evolvability theory is, among other things, to understand how broad and expressive these functions can be allowed to be while still permitting their efficient evolution.

The following is an abbreviated summary of the basic definitional framework. Let $X = \{0, 1\}^n$ be an n -dimensional space of experiences or *examples* (e.g. in the above instance the expression levels of the proteins), a set C of *functions* (e.g. the functions by which the expression level of each protein is determined in terms of the expression levels of the others), and a set R of *representations* of functions (e.g. the DNA strings of the genes). Also we define an *ideal function* f , which would define for each vector $x \in X$ the *best value* from the viewpoint of the evolving organism. In the current instance, for each combination of expression levels

of the other proteins it would define the ideal expression level of the protein at hand. For simplicity here we discuss only Boolean functions with values in $\{-1, 1\}$. We define a distribution D over X that defines the relative probabilities of the various possible vectors $x \in X$ that can occur. We define the *performance* of a representation r to be the correlation of r with the ideal function f taken over all points in X weighted according to D . Formally, we denote $\text{Perf}_f(r, D) = \mathbf{E}_D[r(x) \cdot f(x)]$. In addition, since the exact performance cannot be efficiently computed in many cases without exponential resources, we define the *empirical performance* $\text{Perf}_f(r, D, s)$ of r on samples of size s . It is a random variable that equals $\frac{1}{s} \sum_{i \leq s} (r(z_i) \cdot f(z_i))$ for $z_1, z_2, \dots, z_s \in X$ chosen randomly and independently according to D . A representation r is good if it is similar to the ideal f , or $\text{Perf}_f(r, D) \geq 1 - \epsilon$ for some small $\epsilon > 0$. An *evolutionary algorithm* is defined by a quadruple $A = (R, \text{Neigh}, \mu, t)$ where:

- R is a set of representations of functions over X ;
- $\text{Neigh}(r, \epsilon)$ is a function that for $r \in R$, equals the *neighborhood* of r , that is, the set of representations into which r randomly “mutates”. For all r and ϵ , $r \in \text{Neigh}(r, \epsilon)$ and $|\text{Neigh}(r, \epsilon)| \leq p_A(n, 1/\epsilon)$ for a fixed polynomial p_A .
- $\mu(r, r_1, \epsilon)$ is a function that for $r \in R$ and $r_1 \in \text{Neigh}(r, \epsilon)$, gives the probability that r “mutates” into r_1 ;
- $t(\epsilon)$ is the function that equals the *tolerance* of A . The tolerance determines the difference in performance that a “mutation” has to exhibit to be considered beneficial (or deleterious). The tolerance is bounded from below by a polynomial in $1/n$ and ϵ .

Functions Neigh , μ , and t all need to be computable by a randomized algorithm in time polynomial in n and $1/\epsilon$. The interpretation here is that for each genome the number of variants, determined by Neigh , that can be searched effectively is not unlimited, because the population at any time is not unlimited, but is polynomial bounded. But a significant number of experiences with each variant must be available so that differences in performance can be detected reliably.

We now describe the basic step of such an evolutionary algorithm, designed to model a step of evolution. For a function f , distribution D , evolutionary algorithm $A =$

*Supported by grants from the National Science Foundation NSF-CCF-04-32037 and NSF-CCF-04-27129.

$(R, \text{Neigh}, \mu, t)$, a representation $r \in R$, accuracy ϵ , and sample size s , the *mutator* $\text{Mu}(f, D, A, r, \epsilon, s)$ is a random variable that takes a value r_1 determined as follows. For each $r' \in \text{Neigh}(r, \epsilon)$, it first computes an empirical value of $v(r') = \text{Perf}_f(r', D, s)$. Let

$$\text{Bene} = \{r' \mid v(r') \geq v(r) + t(\epsilon)\}$$

and

$$\text{Neut} = \{r' \mid |v(r') - v(r)| < t(\epsilon)\}.$$

Then

- (i) if $\text{Bene} \neq \emptyset$ then output $r_1 \in \text{Bene}$ with probability

$$\mu(r, r_1, \epsilon) / \sum_{r' \in \text{Bene}} \mu(r, r', \epsilon);$$

- (ii) if $\text{Bene} = \emptyset$ then output $r_1 \in \text{Neut}$ with probability

$$\mu(r, r_1, \epsilon) / \sum_{r' \in \text{Neut}} \mu(r, r', \epsilon).$$

In this definition a distinction between beneficial and neutral mutations is made as revealed by a set of s experiments. If some beneficial mutations are available, one is chosen according to their relative probabilities assigned by μ . If none is available then one of the neutral mutations is chosen according to their relative probabilities assigned by μ . Since in our definition we insist that for all r and ϵ , $r \in \text{Neigh}(r, \epsilon)$, r will always be empirically neutral, and hence Neut nonempty.

Finally we say that a class of functions C is *evolvable* over distribution D if there is an *evolutionary algorithm* $A = (R, \text{Neigh}, \mu, t)$ that for any starting representation $r_0 \in R$ and any ideal function $f \in C$ will converge *efficiently* to a representation r whose performance is close to the performance of f . Formally, there exist polynomials $s(n, 1/\epsilon)$ and $g(n, 1/\epsilon)$ such that for every $f \in C$, every $\epsilon > 0$, and every $r_0 \in R$, with probability at least $1 - \epsilon$, a sequence r_0, r_1, r_2, \dots , where

$$r_i = \text{Mu}(f, D, A, r_{i-1}, \epsilon, s(n, 1/\epsilon))$$

will have $\text{Perf}_f(r_{g(n, 1/\epsilon)}, D) > 1 - \epsilon$.

The polynomial $g(n, 1/\epsilon)$ upper bounds the number of generations needed for the evolution process. A concept class C is *evolvable* if it is *evolvable* over all distributions by a single evolutionary mechanism. We emphasize this by saying *distribution-independently* evolvable.

As in other computational models, such as Turing Machines, the question of how robust the model is under reasonable variations is an important one. Some results along these lines are known. These include the equivalence of evolvability with fixed tolerance t to evolvability with tolerance that might depend on r (see [Val08] for the definitions).

Initial results [Val08] say that monotone conjunctions are evolvable over the uniform distribution and that the evolvable is a subclass of the class that is learnable by Statistical Queries (SQ), defined earlier by Kearns [Kea98], which is known to be a proper subclass of the PAC learnable. Michael gives an algorithm for evolving decision trees over the uniform distribution that is based on a slightly different notion

of performance [Mic07]. Further, Feldman shows that evolvability is equivalent to learning by a natural restriction of statistical queries [Fel08], referred to as *correlational statistical queries*. A correlational statistical query (or CSQ) is a query for the correlation of a given function g with the unknown target function f . The correlation is measured relative to the distribution D over the domain of the learning problem and equals $\mathbf{E}_{x \sim D}[f(x)g(x)]$. To such a query a CSQ oracle returns an estimate of $\mathbf{E}_{x \sim D}[f(x)g(x)]$ within certain tolerance. For comparison, the general SQ model allows queries that provide estimates of $\mathbf{E}_{x \sim D}[\psi(x, f(x))]$ for any function on labeled examples $\psi : \{0, 1\}^n \times \{-1, 1\} \rightarrow \{-1, 1\}$. This equivalence implies that every concept class known to be SQ learnable is evolvable when the distribution over the domain is fixed. In addition, it was shown that decision lists are not evolvable distribution-independently [Fel08], and hence that the evolvable is a proper subclass of SQ.

Open Problems

The main open problem in this direction is characterizing the power of distribution-independent evolvability. In particular, it is unknown whether conjunctions and low-weight linear threshold functions are evolvable distribution-independently. It is easy to see that both classes are weakly evolvable distribution-independently [Fel08] and hence a possible approach is to design an evolutionary variant of boosting (which may be of independent interest). Known boosting techniques rely heavily on information that is not available to an evolutionary algorithm. Evolutionary algorithms for these basic concept classes that use simple mutation mechanisms and converge fast over wide classes of natural distributions would be of particular interest.

More generally, we think that it is important to seek new learning techniques that rely on evolutionary mechanisms of adaptation. Such techniques might shed new light on evolution as it has occurred on Earth and could also find applications outside of the biological context. Identifying such potential applications is another interesting avenue of research.

References

- [Fel08] V. Feldman. Evolvability from learning algorithms. Manuscript. To appear in *Proceedings of STOC*, 2008.
- [Kea98] M. Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM*, 45(6):983–1006, 1998.
- [Mic07] L. Michael. Evolving decision lists. Manuscript, 2007.
- [Val08] L. G. Valiant. Evolvability. To appear in *Journal of the ACM*, 2008.