Semantic Communication for Simple Goals is Equivalent to On-line Learning*

Brendan Juba[1]** and Santosh Vempala[2]***

[1] MIT CSAIL and Harvard University
Maxwell Dworkin 140
33 Oxford St.
Cambridge, MA 02138
bjuba@alum.mit.edu
[2] Georgia Tech
Klaus Advanced Computing Building 2224
266 Ferst Dr.
Atlanta, GA 30332-0765
vempala@cc.gatech.edu

Abstract. Previous works [11, 6] introduced a model of *semantic communication* between a "user" and a "server," in which the user attempts to achieve a given *goal for communication*. They show that whenever the user can *sense* progress, there exist *universal* user strategies that can achieve the goal whenever it is possible for any other user to reliably do so. A drawback of the actual constructions is that the users are inefficient: they enumerate protocols until they discover one that is successful, leading to the potential for exponential overhead in the length of the desired protocol. Goldreich et al. [6] conjectured that this overhead could be reduced to a polynomial dependence if we restricted our attention to classes of sufficiently simple user strategies and goals. In this work, we are able to obtain such universal strategies for some reasonably general special cases by establishing an equivalence between these special cases and the usual model of *mistake-bounded on-line learning* [3, 15]. This equivalence also allows us to see the limits of constructing universal users based on *sensing* and motivates the study of sensing with *richer kinds of feedback*. Along the way, we also establish a new lower bound for the "beliefs model" [12], which demonstrates that constructions of efficient users in that framework rely on the existence of a common "belief" under which all of the servers in a class are designed to be efficient.

Keywords: semantic communication, on-line learning, feedback models

1 Introduction

The *semantic communication* model [11, 6] was introduced to model the problem of communication in the presence of possible *misunderstanding*. In particular,

* This work is also presented in Chapters 4 and 8 of the first author's thesis [10].
** Supported by NSF Awards CCF-0939370, CCF-04-27129, and CCF-09-64401.
*** Supported by NSF Awards AF-0915903 and AF-0910584.

it was intended to address settings where two computers communicate using communications protocols designed and implemented by different parties. In such settings, the possiblity of incompatibility arises, and so it would be desirable for one (or both) of these computers to utilize a *universal* communications protocol that automatically corrects miscommunication. The main results of these works demonstrated that when a *goal for communication* is fixed in advance, such a universal protocol – that achieves its goal whenever its partner supports such functionality – can often be constructed

Here, we attempt to address one of the main deficiencies of earlier results, specifically of results in the *infinite executions* model [6], reviewed in Sec. 2. To be more precise, the main results constructing universal protocols (such as Thm. 5) relied on *enumerating* all algorithms. We are motivated by the desire for constructions that do not suffer from prohibitive overhead, as conjectured to exist [6]. In the *finite executions* model (e.g., the subject of Juba and Sudan [11]) this overhead can be controlled by assuming that the server was "designed" to permit a typical user protocol to run efficiently with respect to some "beliefs" [12]. The constructions do not give good strategies in the infinite execution model, though, since they do not give a finite bound on the number of errors. [3]

We observe that for a restricted kind of goal and sensing that is viable with respect to a class of simple user strategies, the problem of constructing a universal user from sensing is *precisely* the problem of learning the class of concepts corresponding to the simple strategies in the usual on-line learning model [3, 15] (Thm. 8). Thus, each solution to the on-line learning problem for a concept class yields a generic construction of a universal user from a sensing function that is viable with respect to the corresponding class – allowing us to translate an algorithm for efficiently learning linear threshold functions in Thm. 11 to an efficient strategy that works whenever a linear threshold strategy is viable – and vice-versa, allowing us to also translate the negative results. This settles the conjecture of Goldreich et al. [6], establishing that for natural classes of simple strategies and goals, universal user strategies can achieve polynomial overhead in the description length of the desired strategy. We further establish lower bounds that suggest limits to the power of universal users based on the kind of sensing discussed thus far—between the lower bounds that we obtain from the on-line learning model and the new lower bounds we obtain, basic sensing seems to only be adequate for the construction of *efficient* universal users in very simple settings. But, some natural kinds of richer feedback allow the construction of efficient universal users for correspondingly richer user strategies, and we suggest the exploration of such richer feedback as a next step towards constructing universal users of suitable efficiency.

[3] One of our results supports these notions, though: we show in Sec. 5.1 that in order for an efficient user for a class of servers to exist, there must be a common "belief" among *indistinguishable* servers in the class under which typical users are efficient.

2 Semantic Communication in Infinite Executions

The basic model involves a *system* of three interacting entities, a *user*, a *server*, and an *environment*. Each entity has some internal *state*, and they are each joined by a (two-way) communications channel that also has some fixed state on each *round*. Each entity has a *strategy* that specifies a distribution over new internal states and *outgoing messages* for the following round, given the entity's current state and *incoming messages*. We will generally denote the user strategies by U, server strategies by S, and environment strategies by E, respectively.

Thus, given strategies for each of the entities, the system is modeled by a discrete-time Markov process with a state space Ω. We will refer to the infinite sequence of random variables $\{X_t\}_{t=1}^{\infty}$ where X_t is the state of the system in round t as an *execution*; the execution produced by the interaction between a user strategy U, a server strategy S, and an environment strategy E will be denoted by (E, U, S). An *execution started from state* σ_1 is an execution conditioned on $X_1 = \sigma_1$. We denote the space of internal states of the user, server, and environment by $\Omega^{(\mathtt{u})}$, $\Omega^{(\mathtt{s})}$, and $\Omega^{(\mathtt{e})}$, resp., and for $i, j \in \{\mathtt{u}, \mathtt{e}, \mathtt{s}\}$, the state of the communication channel from i to j is a member of $\Omega^{(i,j)}$. Given a state of the system σ, we will let the respective superscripts denote the projection of σ on to the respective components—e.g., $\sigma^{(\mathtt{u},\mathtt{e})}$ is the user's outgoing message to the environment in σ. We wish to design algorithms for user strategies, to be executed by the user in pursuit of a *goal*:

Definition 1 (Goals and robust achievement). *A* goal *is given by a pair* $(\mathcal{E}, R)$*, consisting of a non-deterministic environment strategy and a referee: A* referee R *is a function taking an (infinite) sequence of environment states to a boolean value; we say that an execution is* successful *if the referee evaluates to* 1*. A* non-deterministic strategy $\mathcal{E}$ *is given by a set of (probabilistic) strategies. If a pair of user and server strategies is successful at* $(\mathcal{E}, R)$ *for all* $E \in \mathcal{E}$ *and all initial states of the execution, we say that the pair* robustly achieves the goal.

The interpretation of the environment's non-deterministic strategy is that the environment adversarially chooses a probabilistic strategy E from the set $\mathcal{E}$, effectively making its non-deterministic choices "up-front," allowing us to (sanely) analyze the resulting probabilistic system.

The algorithmic problem we consider is to compute a user strategy that achieves a fixed goal of communication with a large class of server strategies—a *universal* strategy for the goal:

Definition 2 (Universal user). *A user strategy* U *is* $\mathcal{S}$-universal *with respect to a goal* G *if for every server strategy* $S \in \mathcal{S}$*,* (U, S) *robustly achieves the goal.*

We will focus on universal strategies U for which the period of miscommunication in any execution (E, U, S) is uniformly bounded by a polynomial in a size parameter associated with the states of the system, $\mathtt{sz} : \Omega \to \mathbb{N}$. The size will remain fixed throughout an execution (although the world's states may induce many different sizes associated with the same goal).

We will restrict our attention to goals in which time is divided into *sessions* of a fixed length. This is a special case of the *multi-session goals* of Goldreich et al. [6]; we prefer to consider only the special case here because the classes of user strategies we consider are particularly simple, only generating messages for a fixed number of rounds (initially, just one round). Our decision to only consider the special case will also have the added benefit of simplifying the other definitions we use (namely, the number of errors and the corresponding quantitative aspect of our "sensing functions").[4]

Definition 3 (Fixed length multi-session goals). *A goal $G = (\mathcal{E}, R)$ is said to be a k-*round multi-session goal *if the following hold:*

1. (The environment's states.) *The environment's states are partitioned into k sets, $\Omega_1^{(\mathsf{e})}, \ldots, \Omega_k^{(\mathsf{e})}$. We refer to the elements of $\Omega_1^{(\mathsf{e})}$ as* start-session *states, and the elements of $\Omega_k^{(\mathsf{e})}$ as* end-session *states. In each case, the elements of $\Omega_i^{(\mathsf{e})}$ are a pair consisting of an integer* index *and a* contents.
2. (Starting a new session.) *When in an end-session state, the environment non-deterministically moves to a start-session state with an incremented index; furthermore, this non-deterministic choice is independent of the contents of the end-session state.*
3. (Execution of a session.) *When the environment is in some state $(j, \sigma) \in \Omega_i^{(\mathsf{e})}$ for $i \neq k$, $E(j,\sigma)^{(\mathsf{e})}$ is a distribution over $\Omega_{i+1}^{(\mathsf{e})}$ such that every element in its support has index j. Furthermore, the distribution over contents and messages is independent of the index and environment's actual strategy.*
4. (Compact referee) *There is a* temporal decision function *R' taking end-session states to Boolean verdicts, and R is satisfied with an infinite execution iff R' evaluates to zero at most finitely many times.*

The number of times R' evaluates to 0 *in an execution is* the number of errors.[5]

2.1 Sensing: Implicit Descriptions of Goals in Terms of Feedback

Success at a goal of communication is defined as a function of the environment's states, which are not directly visible to the user. Naturally, it is helpful for the user to have some idea of whether or not its current communication strategy is working—indeed, it is *essential* if the user is to be able to reliably succeed in a single session, and many natural goals of communication allow a user to compute such feedback [10]. Although feedback is not *known* to be essential in any sense in multi-session goals, better feedback seems to allow the design of better user strategies [6]. In particular, in this work (as in previous works on semantic communication) we will focus on a relatively minimal kind of feedback that can be computed by the user during an execution.

[4] NB: the decision of "when to halt" is not at issue here, cf. [10, Chapters 2 and 5].

[5] This is a simplification of the notion of errors used by Goldreich et al. [6] where the referee suspending a decision for too long was also onsidered to be an error.

Definition 4 (Sensing, safety, and viability). *A* sensing function V *is a boolean function of the user's view. Let* $G = (\mathcal{E}, R)$ *be a fixed-length multi-session goal with temporal decision function* R' *and size parameter function* $\mathtt{sz} : \Omega \to \mathbb{N}$*, let* S *be a server strategy, and let* $\mathcal{U}$ *be a class of user strategies. For functions* $B : \mathbb{N} \to \mathbb{N}$ *and* $\epsilon : \mathbb{N} \to [0, 1/3]$*,*

- *We say that* V *is* (B, ϵ)-safe *for* G *w.r.t.* $\mathcal{U}$ *and* S *if* $\forall E \in \mathcal{E}, U \in \mathcal{U}, \sigma_1 \in \Omega$*, whenever* $R'(\sigma_1) = 0$*, then w.p.* $1 - \epsilon(\mathtt{sz}(\sigma_1))$*, either only* $B(\mathtt{sz}(\sigma_1))$ *errors will occur, or for some* $t \leq B(\mathtt{sz}(\sigma_1))$*,* V *evaluates to* 0 *in some state* X_t *of the execution* (E, U, S) *started from state* σ_1*.*
- *We say that* V *is* (B, ϵ)-viable *for* G *w.r.t.* $\mathcal{U}$ *and* S *if* $\exists U \in \mathcal{U}\ \forall E \in \mathcal{E}, \sigma_1 \in \Omega$*, w.p. at least* $1 - \epsilon(\mathtt{sz}(\sigma_1))$*, after* $B(\mathtt{sz}(\sigma_1))$ *rounds,* V *evaluates to* 1 *in every subsequent round in the execution* (E, U, S) *started from state* σ_1*.*

If $\epsilon \equiv 0$*, we say that safety (or viability, resp.) holds* perfectly*, and we may refer to such a sensing function as* B-safe *(*B-viable*, resp.).*

Thus, sensing functions encapsulate goal-specific feedback for solving a communications problem. It has been pointed out (by B. Patt-Shamir [17] and an anonymous reviewer) that the role of a sensing function is analogous to that of a *failure detector* in distributed computing [5, 9]. The main difference is that the feedback provided by a failure detector is generally a tentative set of faulty processes (which is the main obstacle in such settings), whereas sensing provides tentative feedback about success at a problem—for example, Juba and Sudan [11] use an interactive proof system to obtain feedback for the goal of computing a function. Although we will motivate a turn to richer feedback in Sec. 5, the main theorems of Goldreich et al. [6] show this simple type of feedback is sufficient for the construction of universal strategies for many goals:

Theorem 5 (On the existence of universal strategies – [6]). *Let* $G = (\mathcal{E}, R)$ *be a fixed-length goal,*[6] $\mathcal{U}$ *be an* enumerable *set of user strategies,* $\mathcal{S}$ *be a set of server strategies, and* $\epsilon : \mathbb{N} \to [0, 1/3]$ *be such that the following hold:*

1. *There is a sensing strategy* V *s.t.* $\forall U \in \mathcal{U}$*, there is a bounding function* B *s.t.* V *is* (B, ϵ)*-safe with* U *and* $\mathcal{S}$ *for* G*, and* $\forall S \in \mathcal{S}\ \exists U \in \mathcal{U}$ *s.t. for the bounding function* B *associated with* U*,* V *is* (B, ϵ)*-viable with respect to* (U, S)*, Furthermore, the mapping* $U \mapsto B$ *is computable.*
   Let $\mathcal{B}$ *denote the set of bounds that appear in the image of this mapping; that is,* $\mathcal{B} = \{B_i : i \in \mathbb{N}\}$*, where* B_i *is the bound for the* i*th user strategy in* $\mathcal{U}$*.*
2. *One of the following two conditions hold: (a) The viability condition holds perfectly (i.e.,* $\epsilon \equiv 0$*). or (b) For every* i*,* $B_{i+1} < B_i/2\epsilon$*.*

Then there is a $\mathcal{S}$*-universal user strategy* U *s.t.* $\forall S \in \mathcal{S}\ \exists B \in \mathcal{B}$ (U, S) *robustly achieves the goal* G *with* $O(B^2)$ *errors, where the constant in the* O*-notation depends on* S*. Furthermore, if* B*-viability holds and the composition of any* $U \in \mathcal{U}$ *with the sensing and enumeration strategies also resides in* $\mathcal{U}$*, then,* $\forall S \in \mathcal{S}$*, the complexity of* U *is bounded by the complexity of some fixed strategy in* $\mathcal{U}$*.*

[6] Actually, the theorem holds for the broader class of *compact* goals, not defined here.

Generic Users for Goals Implicitly Specified by Sensing Theorem 5 gives a single, "generic" construction of a universal user from a sensing function, which can be applied to a variety of examples of sensing functions yielding universal users [6]. In this work, by contrast, we consider the capabilities and limits of such generic constructions, that is, the capabilities and limits of constructions based on *sensing*: rather than directly describing goals, we will assume that we are given a sensing function for a goal, and so the goal is *implicitly* described by the feedback available to the user and the class of strategies that suffice to achieve good feedback, as guaranteed by the viability condition. We then say that the construction is *generic* when it produces a universal user strategy that achieves any goal given only this information:

Definition 6 (Generic universal user). *For a class of goals in infinite executions $\mathcal{G}$, a class of user strategies $\mathcal{U}$, and functions $B : \mathcal{U} \times \mathbb{N} \to \mathbb{N}$, $s : \mathbb{N} \to \mathbb{N}$ and $v : \mathbb{N} \to \mathbb{N}$, we say that U is a B-*error $(\mathcal{U}, s, v)$-generic universal user for $\mathcal{G}$ *if $\forall G \in \mathcal{G}$, any server S, and any sensing function V that is s-safe with S for G and v-viable with S with respect to $\mathcal{U}$ for G, when U is provided the verdicts of V as auxiliary input, (U, S) robustly achieves G with $\min_{U_S \in \mathcal{U} : U_S \ v-\text{viable with } S} B(U_S, \cdot)$ errors.*

There are two primary differences from the statement of Thm. 5: first, Thm. 5 allows for the bounding functions s and v for sensing to vary with the user strategy, and second, the number of errors incurred by Thm. 5 as stated was allowed to depend (arbitrarily) on the server S, whereas we demand that a generic universal user in the present sense obtains a bound that depends uniformly on the "best" user strategy in $\mathcal{U}$. That said, it turns out that for any enumerable class of user strategies $\mathcal{U}$, and $B(U_i, n) = 3i \max\{s(n), v(n)\}^2$, the proof of Thm. 5 actually constructs a B-error $(\mathcal{U}, s, v)$-generic universal user for any fixed-length goal. (Where U_i denotes the ith strategy in the given enumeration of $\mathcal{U}$.) As suggested, we want user strategies that only make *polynomially* many errors (in the length of the description of a target strategy in $\mathcal{U}$, and the size parameter of the execution). Goldreich et al. [6] showed, however, that a polynomial dependence *cannot* be achieved without *some* restrictions on the class of servers: briefly, servers with *passwords* force any strategy to suffer a number of errors that is exponential in the length of the password, and hence in the length of the description of the target strategy.

Thus, we will consider the problem of constructing a generic universal user that succeeds in a polynomial number of errors, given that it is viable with respect to a simple class of strategies. In particular, in Sec. 4, we show that if the class of user strategies $\mathcal{U}$ in the viability condition is sufficiently simple, then we can efficiently identify a good strategy for the class of one-round multi-session goals; in Sec. 5.1, on the other hand, we will see that even for one-round multi-session goals, we will need richer kinds of feedback to efficiently compute good strategies when $\mathcal{U}$ is not so simple. In both cases, the results will follow from an equivalence to *on-line learning* that we describe in more detail next.

3 On-line Learning is Equivalent to Semantic Communication with One-round Goals

Given that an exponential number of errors in the description length of the desired user strategy is unavoidable *in general*, we would like to know *when* it can be avoided. Specifically, we would like to have some conditions under which we can develop efficient universal user strategies for goals in infinite executions. In this section, we investigate one such set of conditions: we will restrict our attention to multi-session goals of communication in which each round corresponds to a distinct session, and assume that sensing with very good safety and viability is available, in which moreover, the sensing function is viable with respect to some class of simple user strategies. Then, a generic construction of universal users from such sensing functions is equivalent to the design of an on-line learning algorithm, and we will find that generic constructions of universal user strategies exist for a variety of classes of simple user strategies.

The model of on-line learning that we consider was introduced by Bārzdiņš and Frievalds [3]. We assume that a *target concept* or *target function* f is drawn from some a priori fixed class of functions $\mathcal{C}$ and the learning algorithm is run in an infinite sequence of *trials* consisting of the following steps: 1. The algorithm is provided an *instance* $x \in X$ as input. 2. The algorithm produces a *prediction* from Y. 3. The algorithm receives *reinforcement* feedback, indicating whether its prediction equaled $f(x)$. In Littlestone's [15] setting, $X = \{0,1\}^n$ and $Y = \{0,1\}$, and then n is a natural size parameter, and $\mathcal{C}$ is finite, but we only require that a suitable notion of size can be defined for X and $\mathcal{C}$. The main parameter used to evaluate these algorithms is the worst case number of mistakes:

Definition 7 (Mistake bounded learning). *For a given on-line learning algorithm A and a concept class $\mathcal{C}$ with* size parameter $n : \mathcal{C} \to \mathbb{N}$*, and any target concept $f : X \to Y$ for $f \in \mathcal{C}$, let $M_A(f)$ be the maximum, over all sequences of instances $\bar{x} = \{x_i \in X\}_{i=1}^{\infty}$, of the number of trials in which A outputs y such that $y \neq f(x_i)$. We then say that a learning algorithm A has* mistake bound $m : \mathbb{N} \to \mathbb{N}$ *if $\forall n' \in \mathbb{N}$ $\max_{f \in \mathcal{C}: n(f)=n'} M_A(f) \leq m(n')$. If the state of A does not change when the algorithm receives positive feedback, then we say A is a* conservative *algorithm.*

Mistake bounded learning algorithms are easily made conservative.

We now show our main theorem, that the special case of semantic communication introduced here – generic users for one-round multi-session goals with a 1-safe and 1-viable sensing function – is *equivalent* to mistake-bounded learning.

Theorem 8. *Let $\mathcal{G}$ be a class of one-round multi-session goals in which the user's incoming messages on each round are drawn from a set $\Omega^{(\cdot,\mathfrak{u})}$, and its outgoing messages are from the set $\Omega^{(\mathfrak{u},\cdot)}$. Let $\mathcal{U}$ be a class of functions $\{U : \Omega^{(\cdot,\mathfrak{u})} \to \Omega^{(\mathfrak{u},\cdot)}\}$ with a size parameter $n : \mathcal{U} \to \mathbb{N}$. Then a conservative $m(n)$-mistake bounded learning algorithm for $\mathcal{U}$ is a m'-error $(\mathcal{U}, 1, 1)$-generic universal user for $\mathcal{G}$ for error bound $m'(U, n') = m(n(U)) + 1$, and conversely, a m'-error*

$(\mathcal{U}, 1, 1)$-generic universal user for $\mathcal{G}$ for error bound $m'(U, n') = m(n(U))$ is a $m(n)$-mistake bounded learning algorithm for $\mathcal{U}$.

Proof. ($\Rightarrow$:) We suppose we are given a conservative $m(n)$-mistake bounded learning algorithm A for $\mathcal{U}$. We will show that A serves as a generic universal user as follows. Suppose we are given $G \in \mathcal{G}$, a server S, and a sensing function V that is 1-safe with S for G and 1-viable with S with respect to $\mathcal{U}$ for G.

By the definition of 1-viability, there is $U_S \in \mathcal{U}$ s.t. if the user sends the same messages as U_S, after one round V will provide a positive indication on every round. Thus, U_S will correspond to the target concept. Each round of the execution will correspond to a trial for the learning algorithm. Suppose we provide the incoming messages to A as the instance, take the prediction of A as the outgoing messages, and provide the verdict of V on the following round as the reinforcement. In particular, if A sends the same outgoing message as U_S, A will receive a positive indication from the sensing function, which we take as positive feedback. Conversely, if V produces a negative indication, then A must not have sent the same outgoing message as U_S would have sent on the incoming messages in that round. V may also produce positive indications when the outgoing message A sent differs from what U_S would have sent, but since A is conservative, the state of A does not change. Now, since A is a $m(n)$-mistake bounded learning algorithm for $\mathcal{U}$, it only receives negative reinforcement $m(n)$ times in any execution.

Since G is a 1-round multi-session goal, R' evaluates to 0 or 1 on each round; when it evaluates to 0, the 1-safety of V guarantees that either that is the only error that will occur, or that V evaluates to 0 in the current round. V is therefore only allowed to evaluate to 1 when an error occurs once, so our strategy therefore makes at most $m(n) + 1$ errors.

($\Leftarrow$:) Let a target concept $U \in \mathcal{U}$ and any sequence of instances $\bar{x} = \{x_i \in \Omega^{(\mathtt{e},\mathtt{u})} \times \Omega^{(\mathtt{s},\mathtt{u})}\}_{i=1}^{\infty}$ be given. We will show how to embed the corresponding sequence of trials into a one-round multi-session goal with a 1-safe and 1-viable sensing function for some server S.

Consider the following one-round multi-session goal $G_U = (\mathcal{E}, R_U)$: the environment non-deterministically chooses $(\sigma_i^{(\mathtt{e},\mathtt{u})}, \sigma_{i+1}^{(\mathtt{s},\mathtt{u})}) \in \Omega^{(\mathtt{e},\mathtt{u})} \times \Omega^{(\mathtt{s},\mathtt{u})}$ for each round i, and sends $(\sigma_i^{(\mathtt{e},\mathtt{u})}, b)$ to the user and $\sigma_{i+1}^{(\mathtt{s},\mathtt{u})}$ to the server. The temporal decision function R'_U for the referee R_U then is satisfied in session i if the user returns $U(\sigma_i^{(\mathtt{e},\mathtt{u})}, \sigma_i^{(\mathtt{s},\mathtt{u})})$. Let S be the server that forwards the message it received from the environment in the previous round to the user in the current round. Let V_U be the sensing function that returns 1 if the user's message on the ith round is $U(\sigma_i^{(\mathtt{e},\mathtt{u})}, \sigma_i^{(\mathtt{s},\mathtt{u})})$. Note that when the user executes with S, V_U computes R'_U, so V_U is 1-safe with S for G_U. Furthermore, whenever the user sends the same message as $U \in \mathcal{U}$, V_U is trivially satisfied on the following round, so V_U is also 1-viable with S with respect to $\mathcal{U}$ for G_U. We can embed $\bar{x}$ in an execution in the following way: let the execution start from the state where $\sigma^{(\mathtt{e},\mathtt{u})} = x_1^{(\mathtt{e},\mathtt{u})}$, $\sigma^{(\mathtt{s},\mathtt{u})} = x_1^{(\mathtt{s},\mathtt{u})}$, and $\sigma^{(\mathtt{e},\mathtt{s})} = x_2^{(\mathtt{s},\mathtt{u})}$, and suppose that the environment's nonde-

terministic choice for the ith round is $(x_{i+1}^{(\mathtt{e},\mathtt{u})}, x_{i+2}^{(\mathtt{s},\mathtt{u})})$. Then, we can check that in each ith round of this execution, the user receives x_i.

Now, supposing that we are given a m'-error $(\mathcal{U}, 1, 1)$-generic universal user for $\mathcal{G}$ A, for every target concept $U \in \mathcal{U}$, A robustly achieves G_U with $m'(U, n') = m(n(U))$ errors when given the feedback from V_U in an execution with S—in particular, in the execution we constructed for a given sequence of trials $\bar{x}$. By definition of G_U, now, A makes an error in the ith round iff it does not send the same messages as U in that round, so when A is provided the feedback from V_U, it makes at most $m(n(U))$ mistakes in the sequence of trials $\bar{x}$. We now note that V_U computes the same function as the learner's reinforcement, so when A is provided access to the reinforcement instead of A, it still only makes $m(n(U))$ mistakes, as needed.

4 Universal Users from On-line Learning Algorithms

We now exploit Thm. 8 to obtain generic constructions of efficient universal users for one-round multi-session goals. In particular, we show that for a variety of halfspace learning, we can confirm one of the main conjectures of Goldreich et al. [6]: there is a universal strategy for a nontrivial class of servers with polynomial overhead. The particular variant we consider has the feature that, unlike previous algorithms (with the exception of the perceptron algorithm) the number of mistakes does not depend on the size of the examples.

Definition 9 (Linear threshold strategies). *The* class of linear threshold strategies *in n dimensions with b-bit weights, $\mathcal{U}_{\mathsf{LT}(n,b)}$, is as follows. We identify the user's incoming messages with $\mathbb{Q}^n$. Then, for any* weight vector $w \in \{-2^{b+1}+1, \ldots, 2^{b+1}-1\}^n$ *and* threshold $c \in \{-2^{b+1}+1, \ldots, 2^{b+1}-1\}$, *the user strategy that on incoming message $x \in \mathbb{Q}^n$ sends $U_{w,c}(x)$ to the server and environment where $U_{w,c}(x) = 1$ if $\sum_{i=1}^{n} w_i x_i \geq c$ and 0 otherwise is in $\mathcal{U}_{\mathsf{LT}(n,b)}$.*

An algorithm for efficiently learning linear threshold functions with *general* weights was given by Maas and Turán [16], based on a reduction to the problem of *finding feasible points in convex programs given by a separation oracle*:

Definition 10 (Convex feasibility with a separation oracle). *Let a convex set $K \subset \mathbb{R}^n$ be given. For $r \in \mathbb{N}$, we say that K has* guarantee r *if the volume of $K \cap \mathsf{Ball}(0, r)$ is at least r^{-n}. A* separation oracle *for K answers queries of the form $x \in \mathbb{Q}^n$ with "yes" if $x \in K$ and otherwise non-deterministically returns a vector $v \in \mathbb{Q}^n$ and $c \in \mathbb{Q}$ such that $\langle x, v\rangle \geq c$, but that for every $y \in K$, $\langle y, v\rangle < c$. If the longest vector v returned by the separation oracle is ℓ bits, we will say that the oracle is* ℓ-bounded.

Now, we say that an oracle algorithm $A^{(\cdot)}$ solves the search problem of convex feasibility with a ℓ-bounded separation oracle *in time $t(n, \log r, \ell)$ and query complexity $q(n, \log r, \ell)$ if, for any ℓ-bounded separation oracle for a convex body K with guarantee r, $A^{(\cdot)}$ produces a point in K in time $t(n, \log r, \ell)$, and making at most $q(n, \log r, \ell)$ queries to the oracle.*

There are efficient algorithms for solving convex programs in this model, yielding algorithms for learning linear threshold functions. The one by Vaidya [18], and an algorithm based on random walks [4] both make at most $O(n \log r)$ queries.

Actually, the above algorithm(s) were for a different problem than the one we consider here: in their model, the *instance space* was assumed to be b-bit integer points (as opposed to $\mathbb{Q}^n$) while the *linear threshold functions* used arbitrary precision (though poly(b) bits suffice to represent all linear thresholds on the b-bit instance space), and the time and query complexity of their algorithm depended the size of the instances. Although it is clear that we cannot hope to eliminate the dependence of the computation time of the size of the instances, it turns out that the dependence on the size of instances in the mistake bound can be eliminated in our setting, using techniques for solving convex programming problems when the convex set K is not of full dimension [7, 8].

Theorem 11. *Suppose there is an algorithm that solves convex feasibility with a ℓ-bounded separation oracle in time $t(n, \log r, \ell)$ and query complexity $q(n, \log r)$ for polynomials t and q. Then there is a $m(n, b)$-mistake bounded on-line learning algorithm for $\mathcal{U}_{\mathsf{LT}(n,b)}$ running in time $t'(n, \log b, \ell)$ on each trial for a polynomial t' where ℓ is the length in bits of the longest received instance $x \in \mathbb{Q}^n$, and $m(n, b) = O(n \cdot q(n, b + \log n))$.*

Sketch of proof The weight vector and threshold of the function $U_{w,c}$ is an integer point in $[-2^{b+1} + 1, 2^{b+1} - 1]^{n+1}$, which is a convex set, and a counterexample x to a proposed linear threshold (w', c') defines a hyperplane such that either $\langle (w', c'), (x, -1) \rangle \geq 0 > \langle (w, c), (x, -1) \rangle$ or $\langle (w, c), (x, -1) \rangle \geq 0 > \langle (w', c'), (x, -1) \rangle$, and either way $(x, -1)$ and 0 gives us a separating hyperplane.

Thus, we pass our counterexamples to the feasibility algorithm, and the algorithm terminates once it finds some point $(\tilde{w}, \tilde{c})$ s.t. any halfspace of the remaining feasible set not containing $(\tilde{w}, \tilde{c})$ has volume less than the guarantee. Then, if we get another counterexample, the hyperplanes given by our counterexamples define a set containing (w, c) of volume less than the guarantee.

By choosing the guarantee sufficiently small, we will be able to ensure that there is a hyperplane such that all of the points with integer coordinates (including the target (w, c)) lie in this hyperplane; we will then be able to find this hyperplane, and reduce to the problem of finding a feasible point in a lower dimensional space by projecting onto it. After we repeat this process $n+1$ times, (w, c) is uniquely determined.

Algorithms for k-round Goals and Strategies with Larger Message Spaces. We exclusively focused on one-round goals and user strategies with Boolean message spaces here. Naturally, this is because the notion of feedback we obtain from sensing only agrees with the usual notions of feedback in on-line learning for Boolean functions, and conversely, on-line learning usually does not handle stateful "concepts." It turns out, though, that a result due to Auer and Long [2] allows us to slightly relax both of these restrictions, giving efficient algorithms for, e.g., $O(\log \log n)$-round goals or messages of size $O(\log \log n)$.

5 Richer Feedback and the Limitations of Basic Sensing

5.1 Limitations of Basic Sensing

We start by presenting a strong lower bound when the space of messages is large. We obtain this via a lower bound on the number of algorithms that an *oblivious* schedule (including Levin-style enumerations and sampling algorithms) must use to escape from a "bad" set of states whenever a class of servers does not have a common prior under which escaping the bad set is easy. We then refine it to handle adaptive algorithms under the assumption that executions with servers in their respective collections of bad states produce indistinguishable user views. The key definition enabling us to state these theorems captures subsets of states of the execution that are hard for typical users to escape:

Definition 12 (Effectively closed). *For a non-deterministic environment strategy $\mathcal{E}$ with size parameter $\mathtt{sz} : \Omega \to \mathbb{N}$, a server S, a distribution over user strategies P, $t : \mathbb{N} \to \mathbb{N}$, and $\gamma : \mathbb{N} \to [0,1]$, we say that the set of server and environment states $\Theta \subseteq \Omega^{(\mathtt{s})} \times \Omega^{(\mathtt{e})}$ is (t,γ)-*effectively closed with respect to *P if, $\forall (\sigma^{(\mathtt{s})}, \sigma^{(\mathtt{e})}) \in \Theta, t \leq t(\mathtt{sz}(\sigma))$ the probability that, for a user strategy U drawn from P, $(X_t^{(\mathtt{s})}, X_t^{(\mathtt{e})}) \in \Theta$ is at least $1-\gamma(\mathtt{sz}(\sigma))$ in the execution (E,U,S) started from σ (for the initial state $\sigma^{(\mathtt{u})}$ specified by U), where the probability is taken over the choice of U from P and the random evolution of the execution.*

These lower bounds also turn out to justify the features of a model introduced in another attempt to refine the notions of semantic communication to reduce the overhead: the "beliefs" model of communication [12]. There, it was assumed that a server was designed to be efficient with respect to "typical" users under a given "belief" (prior) distribution; a user strategy for which communication has low overhead whenever a user has beliefs that are close to those of the server designer then exists. Of course, in order for this approach to guarantee low overhead with respect to an entire class of servers, there must be a *common* belief under which *all* of the servers were designed well. Our lower bound establishes that this common belief was essential in a sense: suppose that we wish to achieve some goal that cannot be achieved while the execution is in one of our "bad sets." Then, our lower bounds demonstrate that when the class of servers lacks a suitable common notion of "natural users" under which escaping the bad sets is easy, a universal user cannot be too efficient, and the best possible running time is roughly obtained by sampling from the best common distribution.

Theorem 13. *Let $G = (\mathcal{E}, R)$ be a goal and $\mathcal{S}$ be a class of servers s.t. $\forall E \in \mathcal{E}, S \in \mathcal{S}$ we have designated some set of pairs of states of E and S, $\Theta_{S,E}$. Let $\delta \in [0,1]$ be given. Now, suppose that $\exists (t,\epsilon) \in \mathbb{N} \times [0,1]$ s.t. for every distribution over user strategies from the class $\mathcal{U}$, Q, $\exists E \in \mathcal{E}, S \in \mathcal{S}$ such that $\Theta_{S,E}$ is (t,ϵ)-effectively closed with respect to Q in E. Then, for any sequence of user strategies and running times $(U_1,t_1),(U_2,t_2),\ldots$ s.t. each $t_i \leq t$, $\exists S \in \mathcal{S}, E \in \mathcal{E}$ s.t. if in the execution where the user runs U_1 for t_1 steps, U_2 for t_2 steps, and so on, the first step τ for which $(X_\tau^{(\mathtt{s})}, X_\tau^{(\mathtt{e})}) \notin \Theta_{S,E}$ is at most $\sum_{i=1}^{k} t_i$ with probability at least δ, then $k \geq \frac{1}{\epsilon(1+1/\delta)}$.*

Proof. In a zero-sum game between a *"user"* player and a *"server/environment"* player, in which the strategy sets are $\mathcal{U}$ and $\mathcal{S} \times \mathcal{E}$, and the payoff of U with (S, E) is given by the maximum probability, over executions starting from initial states from $\Theta_{S,E}$, that the execution exits $\Theta_{S,E}$ in t steps, our assumption on distributions over $\mathcal{U}$ means that the server/environment player always has a good strategy. Loomis' Theorem then yields that there is some distribution $\tilde{Q}$ over $\mathcal{S} \times \mathcal{E}$ such that when any user strategy $U_1 \in \mathcal{U}$ that is run for $t_1 \leq t$ steps with a server and environment pair (S, E) drawn from $\tilde{Q}$ and started in any state of $\Theta_{S,E}$, the probability that the execution (E, U_1, S) enters a state σ such that $(\sigma^{(\mathtt{s})}, \sigma^{(\mathtt{e})}) \notin \Theta_{S,E}$ is at most ϵ.

It then follows by induction on k that, given that the execution never entered a state σ such that $(\sigma^{(\mathtt{s})}, \sigma^{(\mathtt{e})}) \notin \Theta_{S,E}$ during the runs of $U_1, \ldots, U_{k-1}$, during the t_k step run of U_k, the probability that the execution enters such a state σ is at most $\frac{\epsilon}{1-k\epsilon}$. Therefore, while $k\epsilon < 1$, a union bound gives a total probability of exiting $\Theta_{S,E}$ in the first k runs of at most $\frac{k\epsilon}{1-k\epsilon}$. In particular, some (S^*, E^*) in the support of $\tilde{Q}$ must give $(U_1, t_1), \ldots, (U_k, t_k)$ probability at most $\frac{k\epsilon}{1-k\epsilon}$ of exiting $\Theta_{S^*,E}$. Thus, if we exit $\Theta_{S^*,E}$ with probability at least δ by the end of the kth run, this requires $k \geq \frac{1}{\epsilon(1+1/\delta)}$, as needed.

We can extend Theorem 13 to cover adaptive algorithms, given that the servers generate indistinguishable views so long as they remain in the bad states. The key point is that in this case, the algorithm generates a schedule nearly independently of the actual server, essentially reducing to the earlier analysis.

Corollary 14. *Let G, $\mathcal{U}$, $\mathcal{S}$, sets of states $\Theta_{S,E}$ for each $E \in \mathcal{E}$ and each $S \in \mathcal{S}$, and $\delta \in [0, 1]$ be given as in Theorem 13. Suppose that $\exists E \in \mathcal{E}$ s.t. for every distribution Q over $\mathcal{U}$, $\exists S \in \mathcal{S}$ s.t. $\Theta_{S,E}$ is (t, ϵ)-effectively closed with respect to Q in E. Suppose further that $\forall U \in \mathcal{U}, S_1 \in \mathcal{S}, S_2 \in \mathcal{S}, (\sigma_1^{(\mathtt{s})}, \sigma_1^{(\mathtt{e})}) \in \Theta_{S_1,E}$, $\exists (\sigma_2^{(\mathtt{s})}, \sigma_2^{(\mathtt{e})}) \in \Theta_{S_2,E}$ s.t. the distribution over user views in the first t steps of the execution (E, U, S_1) started from a state $(\sigma_1^{(\mathtt{e})}, \sigma^{(\mathtt{u})}, \sigma_1^{(\mathtt{s})})$ is γ-statistically close to the user view in the first t steps of the execution (E, U, S_2) started from the state $(\sigma_2^{(\mathtt{e})}, \sigma^{(\mathtt{u})}, \sigma_2^{(\mathtt{s})})$. Then for any algorithm U that on each step either starts running a new strategy from $\mathcal{U}$ from its initial state or continues running the same strategy from $\mathcal{U}$ for up to at most t steps, $\exists S \in \mathcal{S}$ s.t. if U reaches a state σ s.t. $(\sigma^{(\mathtt{s})}, \sigma^{(\mathtt{e})}) \notin \Theta_{S,E}$ w.p. $\geq \delta$ by running up to k strategies from their initial states, then $k \geq \frac{1}{\epsilon(1+1/\delta)+\gamma/\delta}$.*

Note that we can also apply Corollary 14 to the case where the sets $\Theta_{S,E}$ are chosen to be states of the execution where a given sensing function fails. This will allow us to obtain lower bounds on the performance of any user strategy that uses such a sensing function. Recall that Goldreich et al. [6] showed simple conditions implied an exponential lower bound on the number of errors made by universal users for classes including password-protected servers. Now, for the case of one-round multi-session goals, we know that a lower bound on the number of *rounds* before the referee is satisfied translates into a lower bound on the number

of errors. In this case, we obtain lower bounds with respect to many other classes of user strategies with large message spaces.

Theorem 15. *Let $\mathcal{U}$ be a class of stateless user strategies computing functions $U : X \to Y$ s.t. for every outgoing message y and incoming message x, some $U \in \mathcal{U}$ satisfies $U(x) = y$. Let $G = (\mathcal{E}, R)$ be a one-round goal in which the environment non-deterministically selects an infinite sequence of elements of X, $\mathcal{E} = \{E_{\bar{x}} : \bar{x} = \{x_i \in X\}_{i=1}^{\infty}\}$, s.t. each ith session consists of $E_{\bar{x}}$ sending x_i to both the user and server. The referee's temporal decision function R' is satisfied iff the server receives a message consisting of "1" from the server. Now, let the class of servers $\mathcal{S} = \mathcal{S}(\mathcal{U})$ be s.t. $\forall U \in \mathcal{U}$, $\exists S_U \in \mathcal{S}(\mathcal{U})$ s.t. in each round, the server stores the message $x \in X$ it received from the server until the next round; the server then sends "1" to the user and environment if the user sent a message $y \in Y$ on that round such that $y = U(x)$ for the previous message x that the server received from the environment, and sends "0" to the user and environment otherwise. Then for any user strategy, $\exists S^* \in \mathcal{S}(\mathcal{U})$ s.t. the user strategy makes at least $|Y|/3$ errors w.p. $\geq 1/2$, and at least $|Y|/2$ errors in the worst case.*

It is easy to construct *specific* examples for which learning functions on a message space Y requires an overhead of $|Y| - 1$—Auer and Long[2] describe one such example. Theorem 15, on the other hand, applies to many simple cases of interest, such as linear transformations:

Example 16 (Lower Bound for Linear Transformations). Let $\mathcal{U}$ be the class of linear transformations $A : \mathbb{F}^n \to \mathbb{F}^n$ for some finite field $\mathbb{F}$. Suppose that the instance space is given by $\mathbb{F}^n \setminus \{0\}$. Now, for any nonzero $x, y \in \mathbb{F}^n$ we know that there is some $A_{x,y}$ such that $A(x) = y$. So, Thm. 15 shows that any on-line learning algorithm makes at least $(|\mathbb{F}|^n - 1)/2$ mistakes in the worst case.

We can also use Thm. 8 directly to recover impossibility results for learning *Boolean* functions. Angluin [1] noted that an efficient mistake-bounded learning algorithm gives an efficient PAC-learning algorithm, so negative results for efficient *PAC-learning* also translate to negative results for generic universal users, and so even most natural classes of *Boolean* strategies do not have efficient universal users under cryptographic assumptions (cf. [13, 14]).

5.2 Richer Feedback

Thus, Thm 8 gives a reasonable picture of which classes of strategies we can efficiently learn generically from basic sensing – i.e., with success/fail feedback – and which classes we cannot. Unfortunately, the boundary falls short of where we would like—we can only learn strategies with very small message spaces, and under standard cryptographic assumptions, even then only for fairly simple classes of user strategies.

Recall that our motivation for focusing on this notion of sensing was that we had results, such as Thm. 5, effectively saying that whenever sensing was possible, it was feasible to achieve a goal with any helpful server. As we are

primarily interested in user strategies that do not experience such overhead as that suffered by these constructions, though, we find that we are strongly motivated to investigate some notions of *stronger* feedback (that may not always be available). That is, we view negative results showing that $(\mathcal{U}, 1, 1)$-generic universal users cannot be mistake-efficient and/or time-efficient as limitations of *basic sensing*, and so we seek alternative notions of sensing that do not suffer these limitations. For example, Auer and Long [2] showed how some useful, richer kinds of feedback can be simulated given only basic sensing, but only if the feedback is still limited in the sense that it can be simulated by a logarithmic number of queries; but if these kinds of feedback are directly available, then since we don't need to simulate the feedback, we don't experience this overhead.

Example 17 (Efficient Universal Linear Transformation Strategies from Richer Sensing). Consider the class of user strategies computing linear transformations $A : \mathbb{F}^n \to \mathbb{F}^n$ for some finite field $\mathbb{F}$, as considered in Example 16. There, we saw that given only basic sensing, any generic universal strategy experiences at least $(|\mathbb{F}|^n - 1)/2$ errors for one-round multi-session goals, where Auer and Long's technique yields a universal strategy making $\tilde{O}(|\mathbb{F}|^n)$ errors.

Suppose now that we had richer sensing feedback, that not only provided positive or negative indications, but on a negative indication also provided some index $i \in \{1, \ldots, n\}$ such that if on the previous round we received an incoming message vector $x \in \mathbb{F}^n$ and responded with $y \in \mathbb{F}^n$, a viable linear transformation strategy A would not have responded with $(A(x))_i = y_i$. Then, e.g., for $\mathbb{F}_2$, we could use Gaussian elimination to learn each ith row of a viable linear transformation on $\mathbb{F}_2^n$ in n mistakes, for n^2 mistakes (and time $O(n^3)$ per round) overall. Auer and Long's technique can then also be used to simulate access to $(A(x))_i$ over $\mathbb{F}_q$ for $q > 2$ with an overhead of $\tilde{O}(q)$ mistakes, thus allowing us to use essentially the same learning algorithm over $\mathbb{F}_q$. As long as the field size is still small, this yields polynomial error and polynomial time bounded universal strategies, in contrast to the exponential lower bound of Example 16.

Similarly, if the sensing function also told us that the user's messages in some ith round of the previous session was unsatisfactory, then this feedback would enable us to construct efficient universal users for k-round multi-session goals, given that there are stateless viable user strategies such that a time and mistake efficient on-line learning algorithm for the class of user strategies when restricted to any single round (even if k is polynomially large).

Conclusion These observations suggest the following program for future work. In order to obtain flexible protocols for interesting goals with low overhead, we could first try to identify what kind of feedback is available in those goals, and second, try to determine which classes of strategies can be efficiently learned from such feedback. The hope is that with rich enough feedback, reasonably strong classes of strategies can be learned.

Acknowledgments

This direction was motivated by conversations with Leslie Kaelbling and Leslie Valiant. We thank Oded Goldreich for insightful comments that improved the presentation. Finally, we thank the anonymous referees for their comments.

References

1. Angluin, D.: Queries and concept learning. Mach. Learn. 2(4), 319–342 (1988)
2. Auer, P., Long, P.M.: Structural results about on-line learning models with and without queries. Mach. Learn. 36(3), 147–181 (1999)
3. Bārzdiņš, J., Freivalds, R.: On the prediction of general recursive functions. Soviet Math. Dokl. 13, 1224–1228 (1972)
4. Bertsimas, D., Vempala, S.: Solving convex programs by random walks. J. ACM 51(4), 540–556 (2004)
5. Chandra, T. D., Toueg, S.: Unreliable failure detectors for reliable distributed systems. J. ACM 43(2), 225–267 (1996)
6. Goldreich, O., Juba, B., Sudan, M.: A theory of goal-oriented communication. Tech. Rep. TR09-075, ECCC (2009)
7. Grötschel, M., Lovász, L., Schrijver, A.: Geometric methods in combinatorial optimization. In: Pulleybank, W.R. (ed.) Proc. Silver Jubilee Conf. on Combinatorics. pp. 167–183. Progress in Combinatorial Optimization, Academic Press, New York (1984)
8. Grötschel, M., Lovász, L., Schrijver, A.: Geometric algorithms and combinatorial optimization. Springer, New York, second edn. (1993)
9. Jayanti, P., Toueg, S.: Every problem has a weakest failure detector. In: 27th PODC (2008)
10. Juba, B.: Universal Semantic Communication. Ph.D. thesis, MIT (2010)
11. Juba, B., Sudan, M.: Universal semantic communication I. In: 40th STOC (2008)
12. Juba, B., Sudan, M.: Efficient semantic communication via compatible beliefs. In: 2nd Innovations in Computer Science (2011)
13. Kearns, M., Valiant, L.: Cryptographic limitations on learning Boolean formulae and finite automata. J. ACM 41, 67–95 (1994)
14. Kharitonov, M.: Cryptographic hardness of distribution-specific learning. In: 25th STOC. pp. 372–381 (1993)
15. Littlestone, N.: Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. Mach. Learn. 2(4), 285–318 (1988)
16. Maass, W., Turán, G.: How fast can a threshold gate learn? In: Hanson, S.J., Drastal, G.A., Rivest, R.L. (eds.) Computational learning theory and natural learning systems: Constraints and prospects, vol. 1, pp. 381–414. MIT Press, Cambridge, MA (1994)
17. Patt-Shamir, B.: Personal communication (2010)
18. Vaidya, P.M.: A new algorithm for minimizing convex functions over convex sets. Mathematical Programming 73(3), 291–341 (1996)