The Menu Bar

- Administrivia:
  - Project-p?

- Can we beat the Gold standard?
  - Review of the framework
  - Various stochastic extensions
  - Modern learning theory & sample size
    - Gold results still hold!
  - Learning by setting parameters: the triggering learning algorithm
The problem

- From finite data, induce infinite set
- How is this possible, given limited time & computation?
- Children are not told grammar rules

Ans: put constraints on class of possible grammars (or languages)

To review: the Gold framework

- Components:
- **Target language** $L_{gt}$ or $L_t$ (with target grammar $g_t$), drawn from hypothesis family $H$
- **Data (input) sequences** $D$ and texts $t; t_n$
- **Learning algorithm** (mapping) $A$; output hypothesis after input $t_n A (t_n)$
- **Distance metric** $d$, hypotheses $h$
- **Definition of learnability**:
  $$d(g_t, h_n) \rightarrow_{n \rightarrow \infty} 0$$
Framework for learning

1. **Target Language** $L_t \in L$ is a target language drawn from a class of possible target languages $L$.

2. **Example sentences** $s_i \in L_t$ are drawn from the target language & presented to learner.

3. **Hypothesis Languages** $h \in H$ drawn from a class of possible hypothesis languages that child learners construct on the basis of exposure to the example sentences in the environment.

4. **Learning algorithm** $A$ is a computable procedure by which languages from $H$ are selected given the examples.

Some details

- Languages/grammars – alphabet $\Sigma^*$
- Example sentences
  - Independent of order
  - Or: Assume drawn from probability distribution $\mu$ (relative frequency of various kinds of sentences) – eg, hear shorter sentences more often
  - If $\mu \in L_t$, then the presentation consists of positive examples, o.w.,
  - examples in both $L_t$ & $\Sigma^* - L_t$ (negative examples), i.e., all of $\Sigma^*$ (“informant presentation”)
Learning algorithms & texts

- \( A \) is a mapping from the set of all finite data streams to hypotheses in \( H \).
- Finite data stream of \( k \) examples \( (s_1, s_2, \ldots, s_k) \).
- Set of all data streams of length \( k \),
  \[ D^k = \{(s_1, s_2, \ldots, s_k) | s_i \in \Sigma^*\} = \left(\Sigma^*\right)^k \]
- Set of all finite data sequences \( D = \bigcup_{k>0} D^k \) (enumerable), so:
  \[ A : D \rightarrow H \]
  - Can consider \( A \) to flip coins if need be

If learning by enumeration: The sequence of hypotheses after each sentence is \( h_1, h_2, \ldots \).
Hypothesis after \( n \) sentences is \( h_n \).

ID in the limit - dfns

- **Text** \( t \) of language \( L \) is an infinite sequence of sentences of \( L \) with each sentence of \( L \) occurring at least once ("fair presentation")
- Text \( t_n \) is the first \( n \) sentences of \( t \)
- **Learnability**: Language \( L \) is learnable by algorithm \( A \) if for each \( t \) of \( L \) if there exists a number \( m \) s.t. for all \( n > m, A(t_n) = L \)
- More formally, fix distance metric \( d \), a target grammar \( g_t \), and a text \( t \) for the target language. Learning algorithm \( A \) identifies (learns) \( g_t \) in the limit if
  \[ d(A(t_k), g_t) \rightarrow 0 \text{ as } k \rightarrow \infty \]
Convergence in the limit

\[ d(g_t, h_n) \to_{n \to \infty} 0 \]

- This quantity is called generalization error
- Generalization error goes to 0 as # of examples goes to infinity
- In statistical setting, this error is a random variable & converges to 0 only in probabilistic sense (Valiant – PAC learning)

\[ \varepsilon \text{-learnability & “locking sequence/data set”} \]

Ball of radius \( \varepsilon \)

Locking sequence:
If (finite) sequence \( l_\varepsilon \) gets within \( \varepsilon \) of target & then it stays there
Locking sequence theorem

- **Thm 1** (Blum & Blum, 1975, $\epsilon$ version)
  If $A$ identifies a target grammar $g$ in the limit, then, for every $\epsilon > 0$, $\exists$ a locking sequence $l_e \in D$ s.t.
  
  (i) $l_e \subseteq L_g$
  (ii) $d(A(l_e), g) < \epsilon$ &
  (iii) $d(A(l_e \cdot \sigma), g) < \epsilon$, $\forall \sigma \in D$, $\sigma \subseteq L_g$

- Proof by contradiction. Suppose no such $l_e$

Proof...

- If no such $l_e$, then $\exists$ some $\sigma_i$ s.t.
  
  $d(A(l \cdot \sigma_i), g) \geq \epsilon$

- Use this to construct a text $q$ on which $A$ will not identify the target $L_g$

- Evil daddy: every time guesses get $\epsilon$ close to the target, we’ll tack on a piece of $\sigma_i$ that pushes it outside that $\epsilon$–ball - so, conjectures on $q$ greater than $\epsilon$ infinitely often
The adversarial parent...

- Remember: $d(A \cdot l \sigma_l, g) \geq \varepsilon$
- Easy to be evil: construct $r = s_1, s_2, \ldots, s_n \ldots$ for $L_g$
- Let $q_1 = s_1$. If $d(A \cdot q_i, g) < \varepsilon$, then pick a $\sigma_{q_i}$ and tack it onto the text sequence,

$$q_{i+1} = q_i \sigma_{q_i} s_{i+1}$$

o.w., $d$ is already too large ($> \varepsilon$), so can leave $q_{i+1}$ sequence as $q_i$ followed by $s_{i+1}$

$$q_{i+1} = q_i s_{i+1}$$

Pinocchio sequence...

Evil daddy sequence
Gold’s theorem

• Suppose $A$ is able to identify the family $L$. Then it must identify the infinite language, $L_{inf}$.
• By Thm, a locking sequence exists, $\sigma_{inf}$
• Construct a finite language $L_{\sigma_{inf}}$ from this locking sequence to get locking sequence for $L_{\sigma_{inf}}$ - a different language from $L_{inf}$
• $A$ can’t identify $L_{\sigma_{inf}}$, a contradiction

Example of identification (learning) in the limit – whether TM halts or not

Dfn of learns: $\exists$ some point $m$ after which (i) algorithm $A$ outputs correct answer; and (ii) no longer changes its answer.
The following $A$ will work:
Given any Turing Machine $M_j$, at each time $i$, run the machine for $i$ steps.
If after $i$ steps, if $M$ has not halted, output 0 (i.e., “NO”), o.w., output 1 (i.e, “Yes”)

Suppose TM halts:

\[
\begin{array}{cccccccc}
1 & 2 & 3 & 4 & 5 & \ldots & m & m+1 \\
\text{NO} & \text{NO} & \text{NO} & \text{NO} & \text{NO} & \ldots & \text{NO} & \text{YES} \\
\end{array}
\]

Suppose TM does not halt:

\[
\begin{array}{cccccccc}
1 & 2 & 3 & 4 & 5 & \ldots \\
\text{NO} & \text{NO} & \text{NO} & \text{NO} & \text{NO} & \ldots & \text{NO} & \text{NO}
\end{array}
\]
Exact learning seems too stringent

• Why should we have to speak perfect French forever?
• Can’t we say “MacDonald’s” once in a while?
• Or what about this:
  • You say potato; I say pohtahto; You say potato; I say pohtahto;...

Summary of learnability given Gold

• With positive-only evidence, no interesting families of languages are learnable
• Even if given (sentence, meaning)
• Even if a stochastic grammar (mommy is talking via some distribution $\mu$)
  • BUT if learner knew what the distribution was, they could learn in this case - however, this is almost like knowing the language anyway
If a parent were to provide true negative evidence of the type specified by Gold, interactions would look like the Osbournes:

Child: me want more.
Father: ungrammatical.
Child: want more milk.
Father: ungrammatical.
Child: more milk!
Father: ungrammatical.
Child: cries
Father: ungrammatical

When is learnability possible?

• Strong constraints on distribution
• Finite number of languages/grammars
• Both positive and (lots of) negative evidence
  • the negative evidence must also be ‘fair’ – in the sense of covering the distribution of possibilities (not just a few pinpricks here and there...)
Positive results from Gold

- Active learning: suppose learner can query membership of arbitrary elts of $\Sigma^*$
- Then DFAs learnably in poly time, but CFGs still unlearnable
- So, does enlarge learnability possibilities – but arbitrary query power seems questionable

Relaxing the Gold framework constraints: toward the statistical framework

- Exact identification $\rightarrow$ $\epsilon$–identification
- Identification on all texts $\rightarrow$ identification only on $> 1 - \delta$ (so lose, say, 1% of the time)
  - This is called a ($\epsilon$, $\delta$) framework
Statistical learning theory approach

- Removes most of the assumptions of the Gold framework –
- It does not ask for convergence to exactly the right language
- The learner receives positive and negative examples
- The learning process has to end after a certain number of examples
- Get bounds on the # of examples sentences needed to converge with high probability
- Can also remove assumption of arbitrary resources: efficient (poly time) [Valiant/PAC]


- Distribution-free (no assumptions on the source distribution)
- No assumption about learning algorithm
- TWO key results:
  1. Necessary & sufficient conditions for learning to be possible at all (“capacity” of learning machinery)
  2. Upper & lower bounds on # of examples needed
Statistical learning theory goes further - but same results

- Languages defined as before:
  \[ 1_L(s) = 1 \text{ if } s \in L, 0 \text{ o.w. (an 'indicator function')} \]
- Examples provided by some distribution \( P \) on set of all sentences
- Distances between languages defined as well by the probability measure \( P \)
  \[ d(L_1 - L_2) = \Sigma_S | 1_{L_1}(s) - 1_{L_2}(s) | P(s) \]
  This is a ‘graded distance’ - \( L_1(P) \) topology

Learnability in statistical framework

Model:
- Examples drawn randomly, depending on \( P \)
- After \( l \) data pts, learner conjectures hypothesis \( h_l \) - note, this is now a random variable, because it depends on the randomly generated data
- Dfn: Learner’s hypothesis \( h_l \) converges to the target \( (1_L) \) with probability 1, iff for every \( \varepsilon > 0 \)
  \[ \text{Prob}[d(h_l, 1_L) > \varepsilon] \to l \to \infty 0 \]
  \( P \) is not known to the learner except through the draws
  (What about how \( h \) is chosen? We might want to minimize error from target...)
Standard P(robably) A(approximately) C(orrect) formulation (PAC learning)

- If $h_l$ converges to the target $1_L$ in a weak sense, then for every $\varepsilon > 0$ there exists an $m(\varepsilon, \delta)$ such that for all $l > m(\varepsilon, \delta)$
  $$\text{Prob}[d(h_l, 1_L) > \varepsilon] < \delta$$

With high probability (> 1-\delta) the learner’s hypothesis is approximately close (within \varepsilon in this norm) to the target language

$m$ is the # of samples the learner must draw

$m(\varepsilon, \delta)$ is the sample complexity of learning

Vapnik- Chervonenkis result

- Gets lower & upper bounds on $m(\varepsilon, \delta)$
- Bounds depend on $\varepsilon, \delta$ and a measure of the “capacity” of the hypothesis space $H$ called VC-dimension, $d$

  $$m(\varepsilon, \delta) > f(\varepsilon, \delta, d)$$

- What’s this $d$??
- Note: distribution free!
VC dimension, ”d”

- Measures how much info we can pack into a set of hypotheses, in terms of its discriminability – its learning capacity or flexibility
- Combinatorial complexity
- Defined as the largest $d$ s.t. there exists a set of $d$ points that $H$ can shatter, and $\infty$ otherwise
- Key result: $L$ is learnable iff it has finite VC dimension ($d$ finite)
- Also gives lower bound on # of examples needed
- Defined in terms of “shattering”

Shattering

- Suppose we have a set of points $x_1, x_2, \ldots, x_n$
- If for every different way of partitioning the set of $n$ points into two classes (labeled 0 & 1), a function in $H$ is able to implement the partition (the function will be different for every different partition) we say that the set of points is shattered by $H$
- This says “how rich” or “how powerful” $H$ is – its representational or informational capacity for learning
Shattering – alternative ‘view’

• $\mathcal{H}$ can shatter a set of points iff for every possible training set, there are some way to twiddle the $h$’s such that the training error is 0

Example 1

• Suppose $\mathcal{H}$ is the class of linear separators in 2-D (half-plane slices)
• We have 3 points. With +/- (or 0/1) labels, there are 8 partitions (in general: with $m$ pts, $2^m$ partitions)
• Then any partition of 3 points in a plane can be separated by a half-plane:
Half-planes can shatter any 3 point partition in 2-D: white=0; shaded =1 (there are 8 labelings)

BUT NOT...

But not 4 points – this labeling can’t be done by a half-plane:

...so, VC dimension for \( H \) = half-planes is 3
Another case: class $\mathcal{H}$ is circles (of a restricted sort)

$$\mathcal{H} = f(x,b) = \text{sign}(x \cdot x - b)$$

Can this $f$ shatter the following points?

Is this $\mathcal{H}$ powerful enough to separate 2 points?

$$\mathcal{H} = f(x,b) = \text{signum}(x \cdot x - b)$$

Same circle can’t yield both + and -
This $H$ can separate one point...

**VC dimension intuitions**

- How many distinctions hypothesis can exhibit
- # of *effective* degrees of freedom
- Maximum # of points for which $H$ is unbiased
Main VC result & learning

• If $\mathcal{H}$ has VC-dimension $d$, then $m(\varepsilon, \delta)$, the # of samples required to guarantee learning within $\varepsilon$ of the target language, 1-\delta of the time, is greater than:

$$\log(2) \left( \frac{d}{4} \log \left( \frac{3}{2} \right) + \log \left( \frac{1}{88} \right) \right)$$

This implies

• Finite VC dimension of $\mathcal{H}$ is necessary for (potential) learnability!
• This is true no matter what the distribution is
• This is true no matter what the learning algorithm is
• This is true even for positive and negative examples
Applying VC dimension to language learning

• For \( H \) (or \( L \)) to be learnable, it must have finite VC dimension

• So what about some familiar classes?

• Let’s start with the class of all finite languages (each \( L \) generates only sentences less than a certain length)

VC dimension of finite languages

• is infinite! So the family of finite languages is not learnable (in \( (\varepsilon, \delta) \) or PAC learning terms)!

• Why? the set of finite languages is infinite - the # of states can grow larger and larger as we grow the fsa’s for them

• It is the # of states that distinguish between different equivalence classes of symbols

• This ability to partition can grow without bound - so, for every set of \( d \) points one can partition - shatter - there’s another of size \( d+1 \) one can also shatter - just add one more state
Gulp!

- If class of all finite languages is not PAC learnable, then neither are:
  - fsa’s, cfg’s,...- pick your favorite general set of languages
  - What’s a mother to do?

- Well: posit a priori restrictions – or make the class $\mathcal{H}$ finite in some way

FSAs with $n$ states

- **DO** have finite VC dimension...
- So, as before, they are learnable
- More precisely: their VC dimension is $O(n \log n)$, $n=$ # states
Lower bound for learning

• If $H$ has VC-dimension $d$ then $m(\epsilon, \delta)$, the # of samples required to guarantee learning within $\epsilon$ of the target language, $1-\delta$ of the time, is at least:

$$m(\epsilon, d) > \log(2) \left( \frac{d}{4} \log \left( \frac{3}{2} \right) + \log \left( \frac{1}{8\delta} \right) \right)$$

OK, smarty: what can we do?

• Make the hypothesis space finite, small, and ‘easily separable’
• One solution: parameterize set of possible grammars (languages) according to a small set of parameters
• We’ve seen the head-first/final parameter
English is function-argument form

Other languages are the mirror-inverse: arg-function

This is like Japanese
English form

Bengali, German, Japanese form
Variational space of languages

<table>
<thead>
<tr>
<th>Language</th>
<th>S-H</th>
<th>C-H</th>
<th>C</th>
<th>Ng</th>
<th>Va</th>
<th>Y1</th>
<th>SR</th>
<th>Ser</th>
<th>Op</th>
<th>Ser</th>
<th>LD</th>
<th>Sc</th>
<th>V2</th>
<th>Wh</th>
<th>Prop</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arabic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dutch</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>English</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>French</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>German</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hindi</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Icelandic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Irish</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Italian</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Japanese</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Malay</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mandarin</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Swedish</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tamil</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Actual (prolog) code for this diff

% parametersEng.pl
% % X-Bar Parameters
specInitial.
specFinal :- \+ specInitial.

headInitial(\_).
headFinal(X) :- \+ headInitial(X).

agr(weak).

% % V2 Parameters
% Q is available as adjunction site
boundingNode(i2).
boundingNode(np).

% % Case Adjacency Parameter
CaseAdjacency. % holds

% % Wh In Syntax Parameter
whInSyntax.

% % Pro-Drop Parameter
no proDrop.

% % X-Bar Parameters
specInitial.
specFinal :- \+ specInitial.

headFinal.
headInitial :- \+ headFinal.
headInitial(X) :- \+ headFinal(X).
headFinal(\_) :- headFinal.

agr(strong).
% % V2 Parameters
% % Subjacency Bounding Nodes
boundingNode(i2).
bindingNode(np).

% % Case Adjacency Parameter
no caseAdjacency.

% % Wh In Syntax Parameter
no whInSyntax.

% % Pro-Drop
proDrop.

Learning in parameter space

- Greedy algorithm: start with some randomized parameter settings
  1. Get example sentence, s
  2. If s is parsable (analyzable) by current parameter settings, keep current settings; o.w.,
  3. Randomly flip a parameter setting & go to Step 1.
More details

• 1-bit different example that moves us from one setting to the next is called a trigger

• Let’s do a simple model – 3 parameters only, so 8 possible languages

Tis a gift to be simple...

• Just 3 parameters, so 8 possible languages (grammars) – set 0 or 1

• Complement first/final (dual of Head 1st)
  • English: Complement final (value = 1)
  • Specifier first/final (determiner on right or left, Subject on right or left)
  • Verb second or not (German/not German)
3-parameter case

1. Specifier first or final
2. Complement (Arguments) first/final
3. Verb 2nd or not

Parameters

Spec 1st

Subject Verb...

specifier (subject NP)

Spec final

Verb Subject...

specifier (subject NP)

6.863J/9.611J Lecture 22 Sp03
Comp(lement) Parameter

- **Comp 1st**
  - NP
  - Verb
  - VP

- **Object Verb**

- **Comp final**
  - VP
  - Verb
  - NP

Verb second (V2)

- Finite (tensed) verb **must** appear in exactly 2nd position in main sentence
Even this case can be hard...

• German: dass Karl da Buch kauft
  (that Karl the book buys)
  Karl kauft das Buch
• OK, what are the parameter settings?
• Is German comp-1\textsuperscript{st}? (as the first example suggests) or comp-last?
• Ans: V2 parameter – in main sentence, this moves verb kauft to 2\textsuperscript{nd} position
Input data – 3 parameter case

- Labels: S, V, Aux, O, O1, O2
- All unembedded sentences (psychological fidelity)
- Possible English sentences:
  S V; S V O1 O2; S Aux V O; S Aux V O1 O2; Adv S V; Adv S V O; Adv S V O1 O2; Adv S Aux V; Adv S Aux V O; Adv S Aux V O1 O2
- Too simple, of course: collapses many languages together...
- Like English and French...oops!

Sentences drawn from target

- Uniformly
- From possible target patterns
- Learner starts in random initial state, 1,...8
- What drives learner?
- Errors
Learning driven by language triggering set differences

A trigger is a sentence in one language that isn’t in the other.

\[ |L_i \setminus L_j| / |L_{\text{target}}| \]

How to get there from here

- transitions based on example sentence
  - Prob(transition) based on set differences between languages, normalized by target language \(|L_{\text{target}}|\) examples (in our case, if \(t=\text{English}, 36\) of them)
Formalize this as...

- A Markov chain relative to a target language, as matrix $M$, where $M(i,j)$ gives the transition probability of moving from state $i$ to state $j$ (given target language strings)
- Transition probabilities based on cardinality of the set differences
- $M \times M$ = probabilities after 1 example step; in the limit, we find $M^\infty$
- Here is $M$ when target is $L_5$ = ‘English’

The Ringstrasse (Pax Americana version)
Markov matrix, target = 5 (English)

\[
\begin{array}{ccccccc}
 & L_1 & L_2 & L_3 & L_4 & L_5 & L_6 & L_7 & L_8 \\
L_1 & \frac{1}{2} & \frac{1}{6} & 1 & & & & & \\
L_2 & & \frac{3}{4} & \frac{1}{12} & \frac{1}{12} & & & & \\
L_3 & & & \frac{1}{12} & \frac{1}{2} & \frac{1}{6} & & & \\
L_4 & & & & \frac{1}{6} & \frac{5}{6} & \frac{2}{3} & \frac{1}{9} & \frac{1}{36} \\
L_5 & & & & & \frac{1}{18} & \frac{1}{12} & \frac{1}{36} & 9 \\
L_6 & & & & & & \frac{1}{18} & \frac{1}{9} & \\
L_7 & & & & & & & \frac{1}{9} & \\
L_8 & & & & & & & & \\
\end{array}
\]