Curriculum-Based Learning With STINT

Joseph O'Sullivan

(Tom "global-traveller" Mitchell)

ML Lunch, May 11 1999

Roadmap

- Curriculum Based Learning
- STINT
- STINT-GEN

Curriculum Based Learning

- Not pedagogical learning (semi motivated by)
- Hypothesis:

  Systems that learn from curricula can learn more difficult tasks than those that don't learn from curricula.
- Goal: understanding learners that encounter many tasks over long periods.

Learning Agents

Properties:

- Fixed sensors
- Predefined actions
- Situated in an environment
- At least one underlying learning mechanism

Goal: Improve performance on tasks

Problem: Labelled examples are **expensive**

Priority: **Reduce** the need for labelled examples

Some definitions

Learning task is $f_i : X_i \to Y_i$

($X_i = X_j = X$ ie *single* domain)

Learned approximation to f_i is $\hat{f}_i$

F is the set of tasks being learned $\{f_1 \ldots f_n\}$

$P(\hat{f}_i)$ is some performance metric over $\hat{f}_i$

- number of examples to learn f_i
- generalization error of f_i

Goal is to minimize some loss over P

more definitions...

Curriculum $\mathbb{C}$ is a pair $\langle S, R\rangle$

S is a sequence of examples drawn from X, along with values for one or more of the functions from F

- jth example is $\langle x_j, y_{j1}, y_{j2}, \ldots y_{jn}\rangle$.
- y_{ji} is $f_i(x_j)$ or the token "?"

R is a set of relations defined over F.

- eg $f_i(x) = f_j(x) \wedge f_k(x)$
- eg correlation between $f_i(x)$ and $f_j(x)$ is 0.7

Intuition:

- S characterizes the individual functions
- R characterizes more general knowledge about how the various functions are related.

and more detail..

Fully labeled curriculum: S contains no unknown target values "?".

Singly-labeled curriculum: each example in S contains the value for exactly one of the target functions from F.

Blocked curriculum: singly labeled curriculum in which all examples of each f_i occur contiguously.

$\hat{f}_i^{\mathbb{C}}$ denote a learned approximation to f_i under curriculum $\mathbb{C}$.

Notion of $P(\hat{f}_i^{\mathbb{C}})$ same as before.

Can compare two curricula $\mathbb{C}_1$ and $\mathbb{C}_2$, to see which is better over P.

...

Can do more things, that aren't really important for this talk.

...

Design Issues

Desirable properties include

- effective use of a curriculum,
- robustness to changes in curriculum,
- ability to incorporate a priori information about which tasks are useful and harmful to other tasks,
- effective with both large and small curricula,
- easy to create good curriculum.

(May well be that no one learner is capable of satisfying all properties simultaneously)

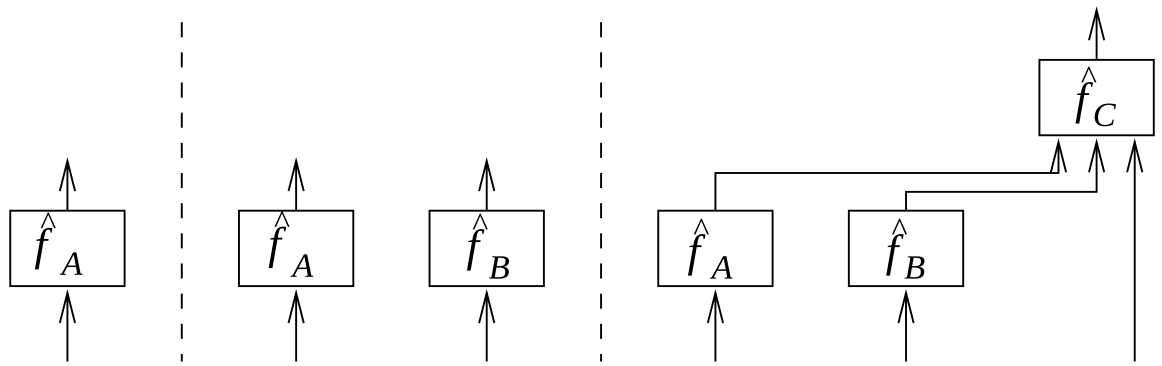
Design Choices

- What underlying learning method?
- How to transfer knowledge?
- How does the data arrive?
- How to use the chosen transfer mechanism?
- Should a priori information be trusted?
- How are tasks related?
- How to decide which previous tasks are related?

Sequences and sTructures in Network Training

STINT is designed for single domain, singly labeled, blocked curricula

STINT learns a *composite* neural network that grows to represent all tasks in the curriculum $\mathbb{C} = \langle S, R \rangle$.



R

Want R to provide information about which tasks will optimize this performance.

When a new task arrives, STINT uses R to select which of the already-learned tasks to use as inputs to the new task.

Ideally: $R =$ assertions of the form *"the utility of f_j as an input to f_i, given the sequence S, is u."*

In practice, input assertions of the form "the utility of f_j as an input to f_i is u."

u = difference between $P(\hat{f}_i(X))$ and $P(\hat{f}_i(X, \hat{f}_j(x)))$.

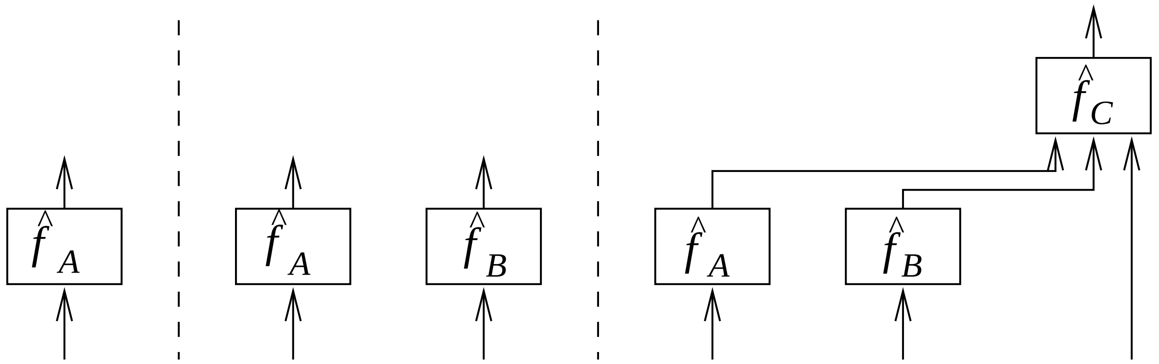
This utility may be interpreted as the expected utility over all possible sequences S.

Inputs and Outputs

Input: a single domain, singly labeled and blocked curriculum $\mathbb{C} = \langle S, R \rangle$.

R must be a set of assertions of the form "the utility of using $\hat{f}_j$ as an input to $\hat{f}_i$ is u."

Output: a trained composite network N, which models $\hat{f}_i^{\mathbb{C}}$ for all $f_i \in \mathbb{C}$.



PseudoCode...

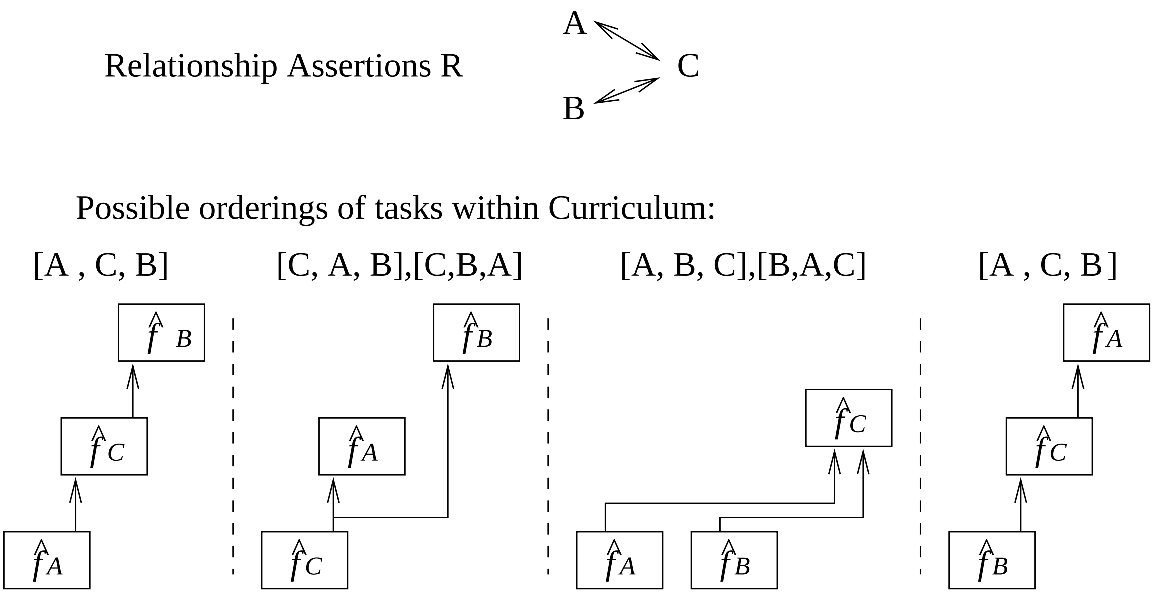
STINT($\mathbb{C}$)
/* **Input**: curriculum $\mathbb{C} = \langle S, R \rangle$. */
$composite \leftarrow$ NIL /* This is the $composite$ network . */
for each example $e \in S$ /* jth example is $\langle x_j, y_{j1}, y_{j2}, \ldots y_{jn} \rangle$ */
  do if e is an example of a new task
    then $T \leftarrow$ name of new task in e
    $composite \leftarrow$ EXPAND$(R, T, composite)$
  Train $composite$ with e
return $composite$
/* **Output**: $composite$ network that provides $\hat{f}_i^{\mathbb{C}}$ for all $f_i \in \mathbb{C}$. */

EXPAND(R, T, N)
/* **Input**: relationship structure R, task name T, composite net N */
$Net \leftarrow$ network T over input domain X with H hidden units
for each edge $(P, T) \in R$
  do if $w(P, T) >$ THRESHOLD
    then $Net \leftarrow$ ADDINPUT(P, Net, N)
$N \leftarrow N \cup Net$;
return N
/* **Output**: composite net N expanded to include learning task T */

ADDINPUT$(P, N, composite)$
/* **Input**: task name P, network N, current $composite$ network */
/* ($composite$ contains a learned approximation to P) */
Extend input layer of N with a new node, X, fully connected to hidden layer
Alias output of P in $composite$ to X
Initialize weights from X to hidden layer to small random values;
return N
/* **Output**: N uses learned task P as an input */

Usage

For a fixed R, the ordering of tasks in the curriculum will determine the composite network created.

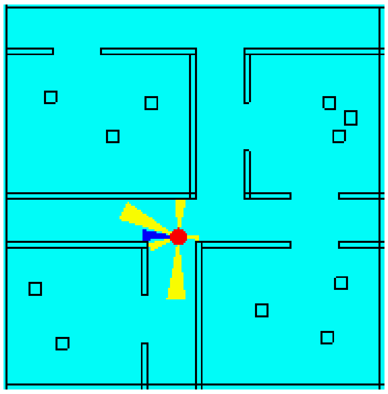


Corresponding composite networks.

Can use penalty terms to control width of input

Can alter learning rates/initial weights to control immediate benefit of tasks.

Syn-Xavier



Task	Description
1	Is the robot facing a door opening onto a pot with a flower?
2	Is the robot facing a door?
3	Is the robot facing a wall?
4	Is the robot in a corridor?
5	Is the robot facing a pot?
6	Is the robot facing a flower?

Setup

Expected :

Generate every *composite network* and corresponding $\mathbb{C}$.

Average performance over all $\mathbb{C} =$

expected performance of a random $\mathbb{C}$.

Optimal : From all $\mathbb{C}$ above, select optimal.

Pessimal : As above, select pessimal.

Hand-coded :

$S = [\text{D, P, F, DFP}]$

$R =$

$$\textsc{Door} \xrightarrow{1} \textsc{Door with Flower in Pot}$$
$$\textsc{Pot} \xrightarrow{1} \textsc{Door with Flower in Pot}$$
$$\textsc{Flower} \xrightarrow{1} \textsc{Door with Flower in Pot}$$

No Curriculum : Don't use a curriculum...

Results

Measuring

$$1 - \frac{\#\text{ examples to learn } f_i \text{ to 80\%, under } \mathbb{C}}{\#\text{ examples to learn } f_i \text{ to 80\%, no } \mathbb{C}}$$

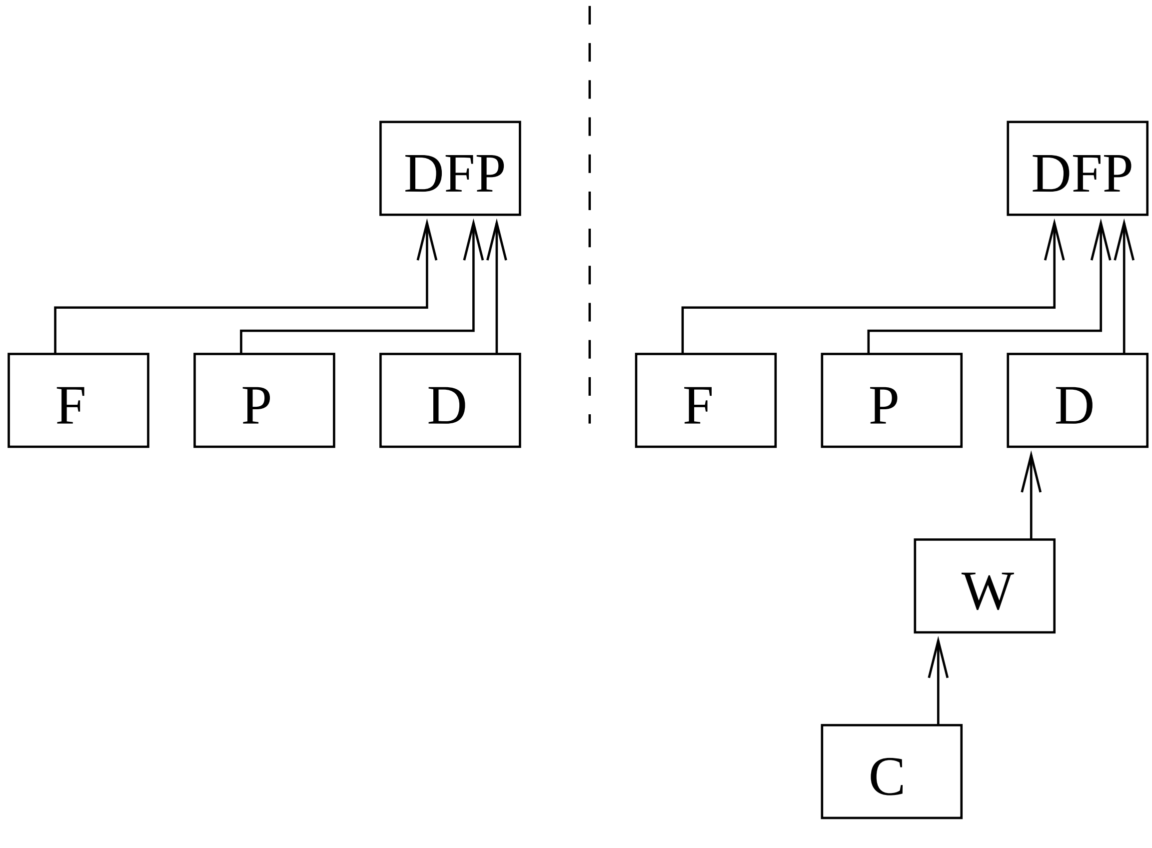
	Previous tasks 80% accurate	Previous tasks 100% accurate
hand coded	78.5%	99.8%
expected	57.8%	92.4%
optimal	84.5%	99.8%
pessimal	0.0%	0.0%

- 10 trials.
- STINT reduces the number of training examples.
- No *bad* tasks so expected does well.
- With perfect knowledge, improvement even greater.

Optimal curriculum

Hand-Coded [D,F,P,DFP]

Optimal [C,W,D,F,P,DFP]



Optimal curriculum in this case outperformed our hand coded curriculum.

STINT-GEN- Automatic generation of curricula

Finding optimal curriculum "difficult".
Given a set of n tasks, the number of possible blocked curricula is equal to the number of permutations of up to n items, or

$$\sum_{k=0}^{n} \binom{n}{k} k!$$

Intuition:

- Generate for the set of tasks a pairwise comparison matrix
- For each pair of tasks (f_x, f_y) in the domain, answer *"Could I be more accurate in my learned approximation of f_x if I knew the true value of f_y"*.
- Examine the graph structure represented by the matrix to derive an approximation to an optimal curriculum.

More detail..

- Task f_x, single immediate predecessor f_k, the benefit due to the curriculum is matrix entry (k, x).
- Task f_x, immediate predecessors $f_1, .., f_k$, the benefit due to the curriculum is function of $\Sigma_{i=\{1..k\}}(i, x)$.
- Benefit is also increased by the benefit of the curriculum to learning each $f_1, .., f_k$.
- We wish to find ordering that maximizes the total sum of these benefits.

Finding this is NP-complete.

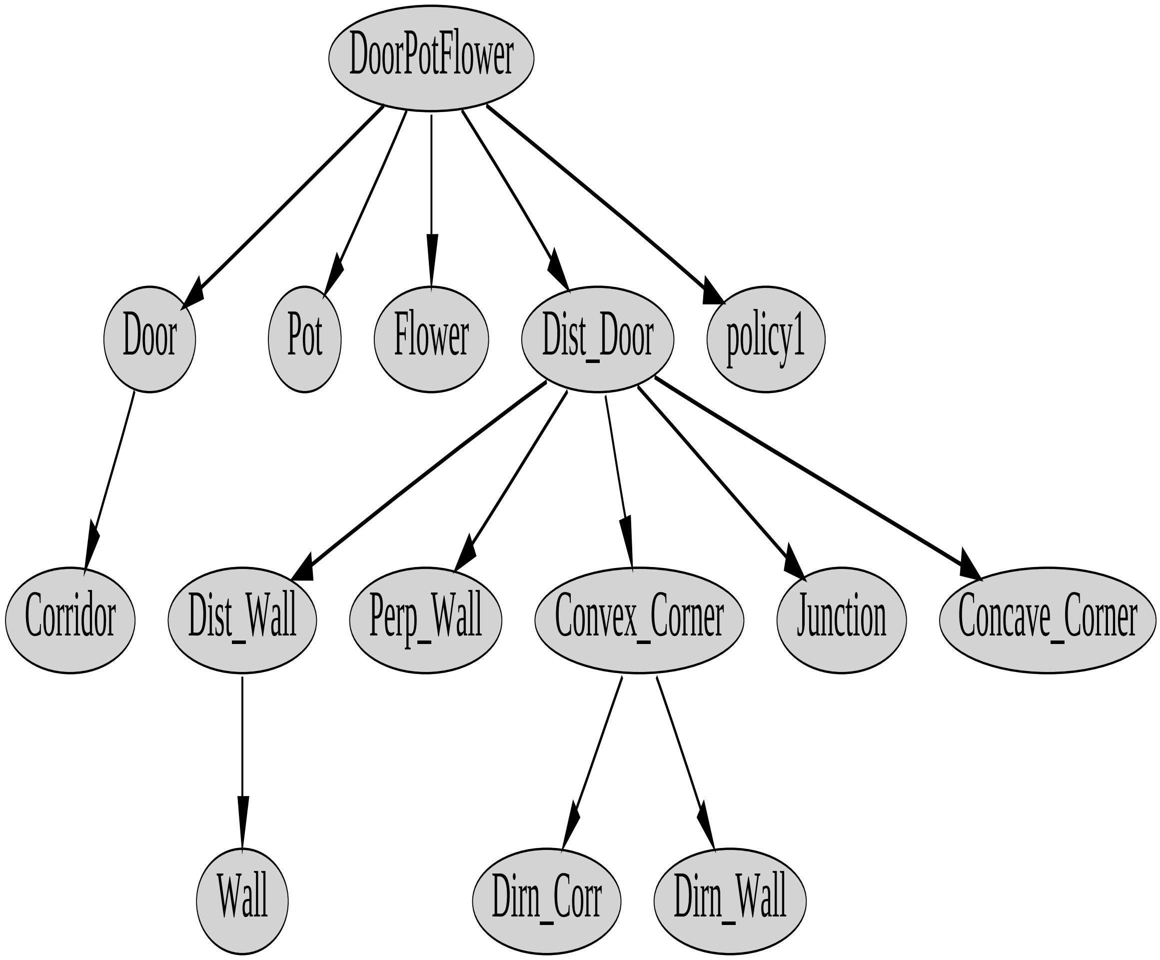
Approximation

A simple greedy algorithm $O(n^3)$, where n is the number of tasks.

- Select the largest entry, $p(i, j)$, in the pairwise matrix,and add that entry as an edge in a graph, along with the nodes f_i and f_j.
- select the next largest entry and add the corresponding nodes.
- Repeat until all nodes are added.

Can actually compute in $O((v + e) \log v)$.

Generated maximal spanning tree



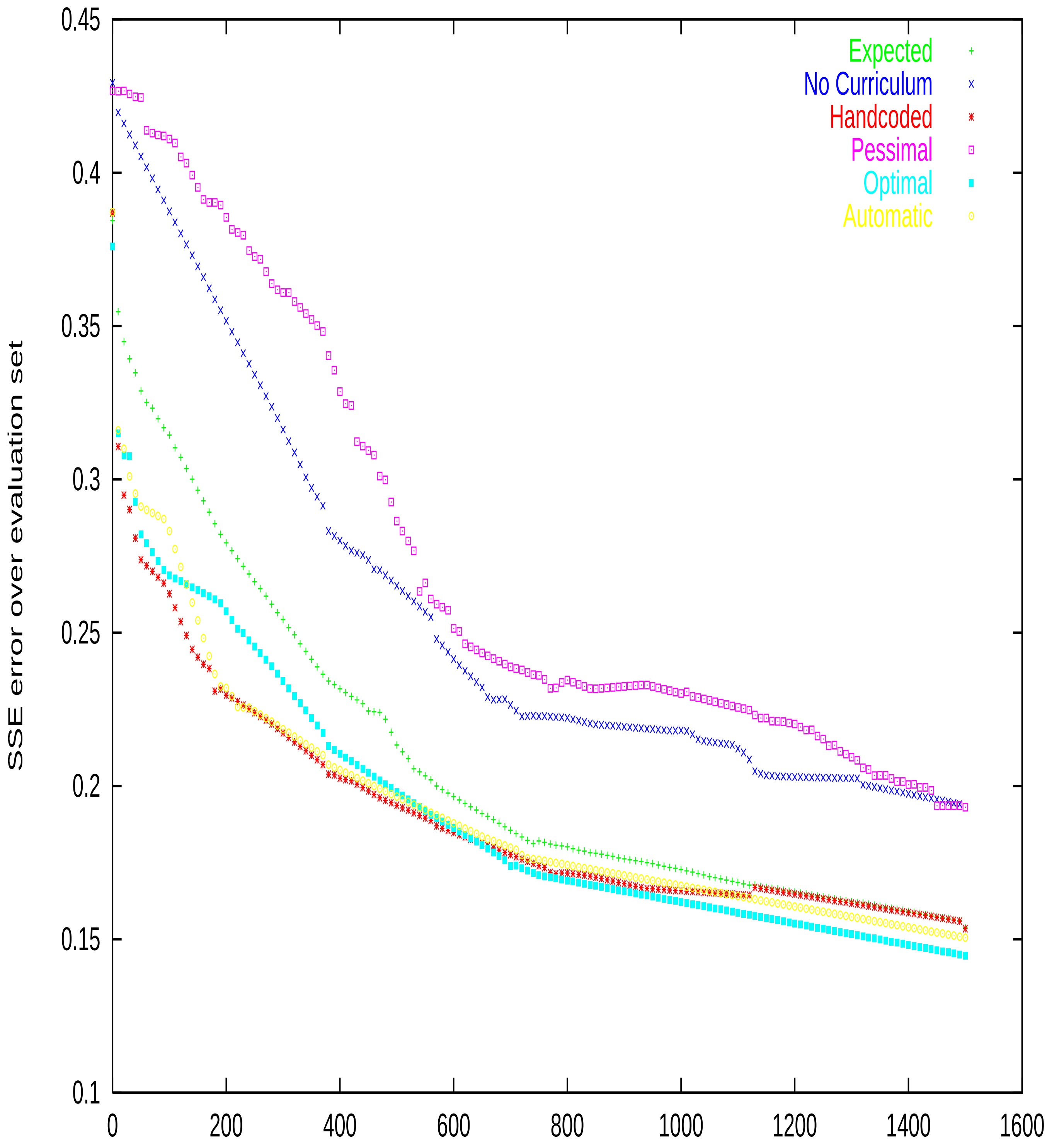
Comments

- Can be extended with initial knowledge from expert.
- Adversarial case $O(|n|)$ within optimal.
- Can use a more expensive algorithm to get tight bound of $\frac{1}{2}$ optimal.
- Can use strongly connected components to improve running time.

Syn-Xavier, more tasks

Description	Range
How far is the nearest wall in front?	0..MAX
Is there a wall perpendicular in front?	0/1
What is the direction of the corridor?	0..16
Is the agent in a junction?	0/1
Is there a concave corner in front?	0/1
Is there a convex corner in front?	0/1
In what dirn runs the wall in front?	0..16
How far to the door??	0..MAX
Follow Corridor policy	0..1
Goto locn in corridor policy	0..1
Follow Wall policy	0..1
Go through door policy	0..1
Goto location in room 1 policy	0..1
Goto location in room 2	0..1
How far to obstacle in front?	0..1
In what dirn is the nearest obstacle?	0..16

Results



SSE error over evaluation set
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0
200
400
600
800
1000
1200
1400
1600
Expected
No Curriculum
Handcoded
Pessimal
Optimal
Automatic

Discussion

- STINT, STINT-GEN kept quite separate.
- In practice, combined approaches, active/online methods
- Using a very specific form of bias, others possible.
- Refinements for recency behavior, knowledge transfered.
- Extending to non-singly labelled, non-blocked curricula.
- May be natural way of eliciting domain knowledge.
- New bias knob for learning systems
- Don't care about underlying learner.
- STINT not a true network construction alg.

Related Stuff

- Ways of inserting bias:
  - procedural
  - representational
- Lifelong learning:
  - Here concerned about sequencing.
  - Making relationships explicit.
- Ordering/Ranking.
  - social sciences, informational retrieval, mathematical economics...

Conclusions

- Curriculum-Based Learner.
- *curriculum* $\mathbb{C}$: data, partial ordering over data, and over the tasks.
- STINT designed to work with singly labeled, blocked curricula

  R assertions "the utility of using $\hat{f}_j(x)$ as an input to the learned model $\hat{f}_i$ is u."
- STINT benefits from curricula.
- STINT-GEN automatically generates good curricula for STINT.